Základná škola Andreja Kmeťa, Ul. M. R. Štefánika 34, Levice



**Asistenčné systémy v automobiloch**

Autor: Ján Mederly

Trieda: VI. E

Školský rok: 2015/2016

Konzultant: Pavol Mederly

Obsah

[Úvod 1](#_Toc449913380)

[Opis vybraných asistenčných systémov 2](#_Toc449913381)

[Airbag 2](#_Toc449913382)

[Antiblokovací systém – ABS 2](#_Toc449913383)

[Regulácia preklzovania kolies – ASR 5](#_Toc449913384)

[Elektronický stabilizačný systém – ESP 5](#_Toc449913385)

[Natáčanie všetkých 4 kolies – 4control 7](#_Toc449913386)

[Brzdový asistent – BAS 7](#_Toc449913387)

[Multikolízne brzdy (asistent primárneho a sekundárneho nárazu) 7](#_Toc449913388)

[Automatické brzdenie pred prekážkou – City safety 8](#_Toc449913389)

[Adaptívny tempomat 8](#_Toc449913390)

[Asistent mŕtveho uhla 9](#_Toc449913391)

[Asistent udržania jazdného pruhu – Lane assist 9](#_Toc449913392)

[Asistent rozpoznávania dopravných značiek – Sign assist 9](#_Toc449913393)

[Parkovacie senzory 9](#_Toc449913394)

[Parkovacia kamera 10](#_Toc449913395)

[Parkovací asistent 10](#_Toc449913396)

[Asistent rozjazdu do kopca 10](#_Toc449913397)

[Asistent zjazdu z kopca 10](#_Toc449913398)

[Kontrola stavu pneumatík 10](#_Toc449913399)

[Adaptívne predné svetlá – Dynamic light assist 11](#_Toc449913400)

[Dažďový senzor 11](#_Toc449913401)

[Vlastný prínos 12](#_Toc449913402)

[Stavba robota 12](#_Toc449913403)

[Diaľkové ovládanie robota 13](#_Toc449913404)

[Asistenčné systémy 14](#_Toc449913405)

[Zhrnutie funkcií robota 17](#_Toc449913406)

[Súčasti systému 18](#_Toc449913407)

[Záver 25](#_Toc449913408)

[Použité zdroje 26](#_Toc449913409)

[Príloha 29](#_Toc449913410)

[Program pre Arduino (na robotovi) 29](#_Toc449913411)

[Program na ovládanie robota (na PC) 35](#_Toc449913412)

# Úvod

Na svete sa vyrába čoraz viac áut a s počtom áut rastie aj počet dopravných nehôd. V dôsledku dopravných nehôd celosvetovo umiera každoročne okolo 1,2 milióna ľudí (1), (2). Už len v Európe v roku 2013 zomrelo pri dopravných nehodách 26 000 ľudí a 199 000 bolo ťažko zranených (3). Väčšina výrobcov automobilov sa snaží toto strašné číslo znížiť.

Významným spôsobom zníženia počtu smrteľných dopravných nehôd sú asistenčné systémy.

Asistenčné systémy pomáhajú vodičovi riadiť auto tak, aby nenastala žiadna nehoda alebo v prípade nehody ochraňujú posádku. Asistenčné systémy v autách sa rozdeľujú do dvoch základných skupín: aktívne a pasívne. (1)

**Aktívne –** sú to také, ktoré nie sú priamo spojené s kolíziou. Snažia sa, aby kolízia ani nenastala. Medzi najčastejšie patria ABS a ESP.

**Pasívne –** sú to také, ktoré ochraňujú posádku, keď dôjde ku kolízii. Medzi najjednoduchšie patria bezpečnostné pásy. Ďalej sú to napr. airbagy.

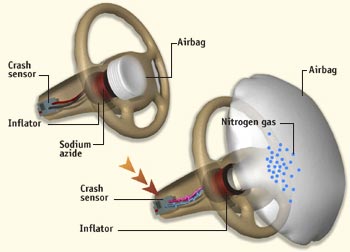
V tejto ročníkovej práci sú opísané vybrané asistenčné systémy. Ako vlastný prínos som spravil robota s niektorými asistenčnými systémami.

# Opis vybraných asistenčných systémov

## Airbag

Pri kolízii sa pred pasažierom nafúkne vak a tak zabráni, aby sa pasažier udrel o jednotlivé časti automobilu.

Pokiaľ dôjde ku kolízii, zaznamenajú ju akcelerometre. Akcelerometre sú senzory, ktoré sledujú zrýchlenie alebo spomalenie. Riadiaca elektronika aktivuje vyvíjač plynu, ktorý je zväčša na báze rozkladu azidu sodného. (Azid sodný je vysoko toxická pevná látka.) Balón sa nafúkne po cca 40 ms (0,04 s). (4)



Obrázok - airbag

## Antiblokovací systém – ABS

Antiblokovací systém (ABS, z nem. Antiblockiersystem, po anglicky Antiskid Brake System) zabraňuje zablokovaniu kolies pri plnom brzdení alebo na klzkom povrchu a zachováva ovládateľnosť vozidla. (5)

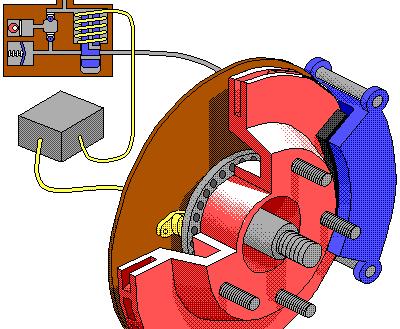
 

Obrázok - s ABS Obrázok - bez ABS

Riadiaca jednotka

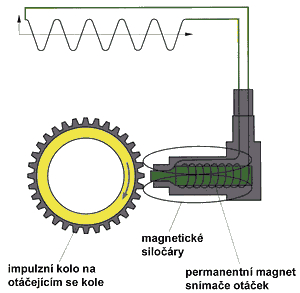
Tlakový ventil

Indukčný merač otáčok



Obrázok - časti ABS

Riadiaca jednotka (šedá krabička) dostane oznam o počte otáčok z indukčného snímača (žltý „štuplík“ pri brzdovom kotúči). Ak riadiaca jednotka dostane signál že koleso je blokované, zníži jeho brzdný tlak pomocou tlakového ventilu (hnedá súčiastka v ľavom hornom rohu) až tak že koleso sa voľne točí. Tento proces sa opakuje 12 – 16 krát za sekundu. (6)



Obrázok - indukčný snímač otáčok

Indukčný snímač otáčok sa skladá z ozubeného kolesa, magnetického jadra cievky a cievky. Podľa toho, či je k magnetu priblížený zub alebo priehlbina, sa mení magnetické pole okolo cievky, v ktorej vzniká indukované napätie. Podľa rýchlosti zmeny indukovaného napätia vieme určiť otáčky kolesa. (7)

Vyskúšal som si aj ja indukovať napätie na cievke pomocou doma vyrobenej cievky a magnetov GEOMAG.

Pomôcky:

* osciloskop Rigol DS1102D,
* cievka vyrobená z medeného drôtu namotaného na prázdne puzdro od cínu,
* magnety zo stavebnice GEOMAG.

Postup:

1. Pripojíme kábliky od cievky na osciloskopové sondy.
2. Z magnetov zo stavebnice GEOMAG postavíme tyč.
3. S tyčou pohybujeme vnútri cievky.



Obrázok - vykonávanie pokusu s indukovaním napätia

|  |  |
| --- | --- |
| \\nsa325\prenos\RP6\IMG_7341.JPG  Obrázok - výstup na osciloskope, keď som nepohyboval magnetom | \\nsa325\prenos\RP6\IMG_7340.JPG  Obrázok - výstup na osciloskope, keď som pohyboval magnetom |

Keď som nepohyboval magnetom, nevytváralo sa napätie. Napätie na osciloskope bolo len 60 mV, ktoré bolo spôsobené elektromagnetickým rušením. Keď som pohyboval magnetom, vytváralo sa napätie. Napätie na osciloskope bolo 780 mV. Keď som mal kratší magnet, tak sa vytváralo menšie napätie. Čím rýchlejšie som pohyboval magnetom, tým viac napätia som vyprodukoval.

Vodič cíti zásah ABS ľahkým pulzovaním brzdového pedálu. Pomocou ABS sa dá bezpečne brzdiť aj vo vysokých rýchlostiach (8).

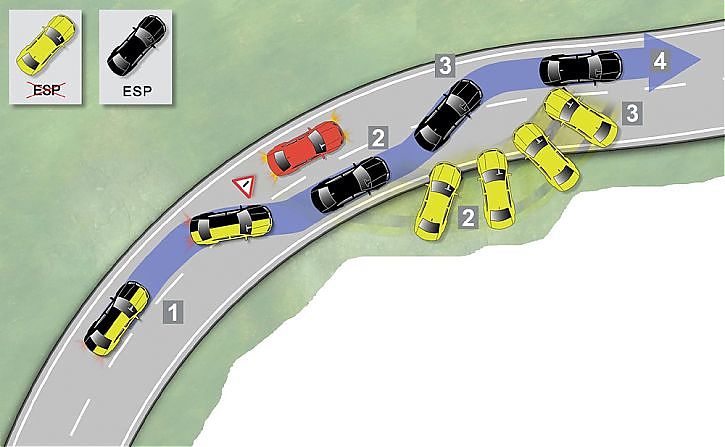
Systém Offroad-ABS v modeli Volkswagen Touareg predstavuje zvláštny algoritmus riadiacej jednotky ABS, ktorý sa automaticky aktivuje až v extrémnom teréne, napríklad na štrku alebo piesku. Pri systéme Offroad-ABS môžu byť kolesá na krátku dobu zablokované, skôr ako systém zníži brzdný tlak. Na povrchu pred kolesom sa tak vytvorí akási zábrana (napríklad zo štrku alebo piesku), ktorá zvýši brzdný účinok. Touareg tak zostáva stále ovládateľný pri výrazne kratšej brzdnej dráhe (8).

## Regulácia preklzovania kolies – ASR

Úlohou ASR (anglicky Acceleration Slip Regulation - z nememeckého Antriebsschlupfregelung (9)) je zabezpečiť prenos hnacej sily z kolies na cestu. Systém ASR spolu so systémom ABS sleduje rýchlosť otáčania jednotlivých kolies. Keď sa koleso z poháňanej nápravy točí rýchlejšie ako koleso z nepoháňanej, znamená to, že sa šmýka. V takom prípade riadiaca jednotka upraví výkon motora tak, aby sa rýchlosť poháňaného kolesa rovnala rýchlosti nepoháňaného. (10)

## Elektronický stabilizačný systém – ESP

Elektronický stabilizačný systém ESP (z [nem](https://sk.wikipedia.org/wiki/Nem%C4%8Dina). Elektronisches Stabilitätsprogramm) pribrzďuje alebo odbrzďuje oddelene zadné a predné kolesá auta a tým auto nie len pri brzdení, ale aj pri zrýchľovaní alebo pri prechádzaní zákruty udržiava mimo šmyku (11).



Obrázok - funkcia ESP (12)

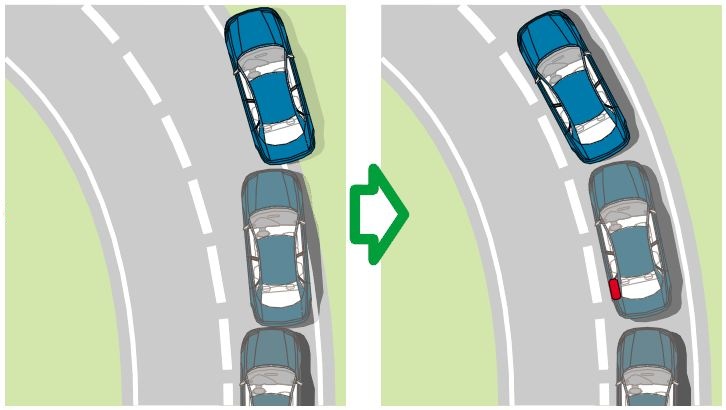
Na obrázku môžeme vidieť, ako funguje ESP. Čierne auto, ktoré má ESP, sa vie bez šmyku vyhnúť červenému autu. Žlté auto, ktoré nemá ESP, pri vyhýbaní dostane šmyk a vyjde von z cesty.

Aby mohol elektronický stabilizačný program na kritické jazdné situácie reagovať, musí vedieť odpoveď na dve základné otázky. aký je želaný smer jazdy a kam vozidlo aj naozaj ide. Na to mu slúžia tieto snímače:

* snímač uhla natočenia volantu,
* snímač tlaku brzdovej kvapaliny,
* snímač polohy plynového pedála,
* snímač otáčok všetkých kolies,
* snímač pozdĺžneho a priečneho zrýchlenia,
* snímač otáčania sa okolo zvislej osi auta.

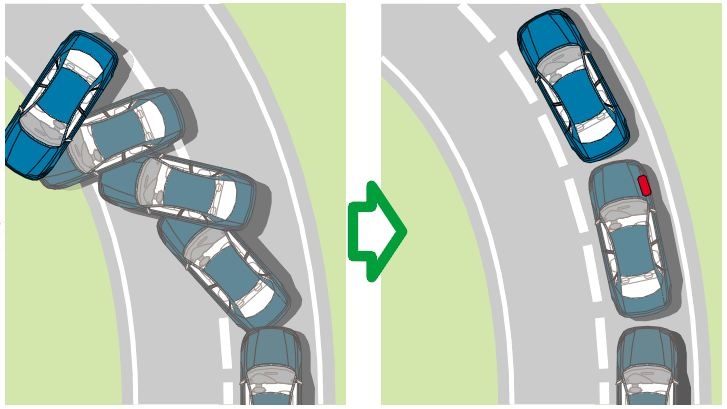
Asistenčný systém zistí, kam vodič vozidlo smeruje, pomocou týchto snímačov: snímač uhla natočenia volantu, snímač tlaku v hlavnom brzdovom valci a snímač polohy plynového pedálu. Aby asistenčný systém vedel, kam auto naozaj ide, poslúžia mu tieto snímače: snímač pozdĺžneho a priečneho zrýchlenia, snímač otáčania sa okolo zvislej osi vozidla a snímač otáčok kolies. Na základe týchto hodnôt riadiaca jednotka porovná želanú dráhu vozidla so skutočnou. Keď sa hodnoty líšia, vyhodnotí situáciu ako kritickú a zasiahne (13).

Teraz idem opísať funkciu ESP pri dvoch možných šmykoch v zákrutách. Pri nedotáčavosti auto nedostatočne zabočí kolesami a chce vyjsť z cesty na vonkajšej strane zákruty. ESP pribrzdí zadné koleso na vnútornej strane zákruty a prípadne zníži výkon motora. Keď ESP pribrzdí toto koleso, vozidlo v ňom nájde „oporný bod“, okolo ktorého sa začne točiť do zákruty.



Obrázok - účinok ESP pri nedotáčavosti

Pri pretáčavosti vozidlo príliš zabočí a chce vyjsť z vozovky na vnútornej strane zákruty. ESP pribrzdí predné vonkajšie koleso a zníži výkon motora. Tým sa zabráni tomu, aby sa vozidlo začalo otáčať do vnútornej strany zákruty.



Obrázok - účinok ESP pri pretáčavosti

„Európska komisia tvrdí, že za posledné dva roky zabránil elektronicky stabilizačný program ESP 33 tisíc dopravným nehodám a zachránil minimálne tisíc ľudských životov.“ (15)

## Natáčanie všetkých 4 kolies – 4control

4control je asistenčný systém, ktorý v rýchlosti do 60 km/h natáča zadné kolesá opačne ako predné, čo spôsobuje oveľa lepšiu ovládateľnosť v meste, menší polomer otáčania. V rýchlostiach nad 60 km/h systém natáča zadné kolesá tak ako predné, čo spôsobuje lepšie držanie stopy. 4control je značka firmy Renault, natáčanie všetkých 4 kolies majú však aj iné značky automobilov (16).

## Brzdový asistent – BAS

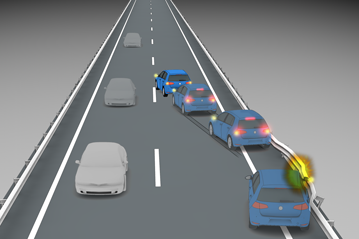
Častou príčinou dopravnej nehody je, že vodič zošliapne brzdový pedál, ale nie dostatočne silno. Brzdový asistent keď zistí, že vodič prudko šliapol na brzdový pedál, zvýši brzdný účinok na maximum, čím sa skráti brzdná dráha (17).

## Multikolízne brzdy (asistent primárneho a sekundárneho nárazu)

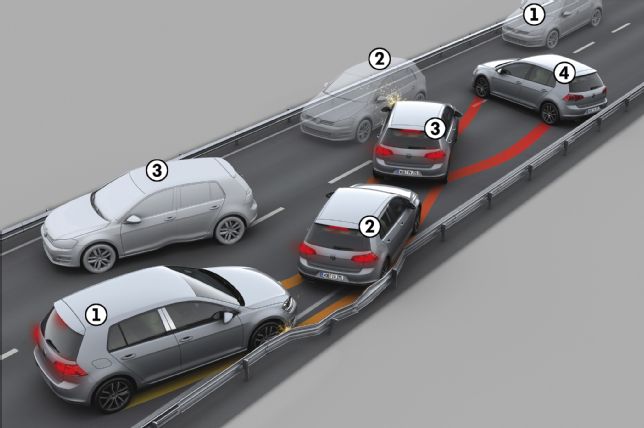
Asistent primárneho a sekundárneho nárazu zabraňuje sekundárnemu nárazu, ktorý zvykne byť oveľa väčší ako primárny náraz. Sekundárny náraz býva omnoho horší, lebo v horšom prípade auto z druhého pruhu narazí v plnej rýchlosti do havarovaného auta, na ktorom už nefungujú použité bezpečnostné prvky napr. airbagy, napínače bezpečnostných pásov atď. V lepšom prípade havarované auto narazí do idúceho s dosť menšou rýchlosťou, ale aj tak to môže byť nehoda s vážnymi dôsledkami. (18)

Podľa štatistík každá štvrtá havária je viacnásobnou haváriou. Ak by bol tento systém 100 percentne nasadený, tak by sa v Európe znížil počet smrteľných úrazov až o 8 percent a počet vážnych zranení až o 4 percentá.

Multikolízne brzdy spôsobia, že keď sa vyšle signál z airbagov, že auto narazilo, tak ihneď začnú brzdy brzdiť. Lane assist spôsobí, že auto nevyjde z jazdného pruhu. Keď auto úspešne zabrzdí, rozsvietia sa výstražné svetlá.



Obrázok - s multikolíznou brzdou



Sekundárny náraz

Primárny náraz

Obrázok - bez multikolíznej brzdy

## Automatické brzdenie pred prekážkou – City safety

City safety je asistenčný systém, s ktorým prišla firma Volvo v roku 2009 a vyrobila ho pre model Volvo XC60. Teraz už tento asistenčný systém má veľa áut. Tento asistenčný systém pomocou radaru zisťuje, či je pred autom prekážka a upozorní na to vodiča – buď grafickým alebo akustickým signálom alebo obidvoma. Ak vodič nereaguje, tak auto samo zabrzdí.

## Adaptívny tempomat

Adaptívny tempomat dodržuje rýchlosť nastavenú vodičom. Keď vidí, že blízko je auto, tak spomalí a udržiava nastavený odstup. Na palubnom počítači sa dá nastaviť, aký odstup má systém dodržiavať.



Obrázok - Asistent mŕtveho uhla

## Asistent mŕtveho uhla

Asistent mŕtveho uhla upozorní na vozidlo, ktoré je v susednom pruhu za vami a vy chcete prejsť do toho pruhu. Radarové snímače integrované v zadnom nárazníku pri rýchlosti nad 10 km/h sledujú oblasť 50 metrov za vozidlom. Ak sa zozadu blíži nejaké vozidlo, aby vás predbehlo, elektronika varuje vodiča rozsvietením kontroliek v spätnom zrkadle (19).



Obrázok - kontrolky v spätnom zrkadle

## Asistent udržania jazdného pruhu – Lane assist

Auto pomocou kamery za čelným sklom sleduje jazdné pruhy. Ak elektronika zaznamená, že auto opúšťa jazdný pruh bez toho, aby vodič zapol smerovku, natočením volantu vráti auto späť do jazdného pruhu (20).

## Asistent rozpoznávania dopravných značiek – Sign assist

Keď sa pozeráme inde ako na cestu (na bilboardy, nové domy) tak neraz si nevšimneme dopravnú značku. Systém sign assist pomocou kamery za čelným sklom upozorní na dopravné značky (21).

## Parkovacie senzory

Parkovacie senzory, ktoré sú v prednom alebo zadnom nárazníku merajú vzdialenosť od prekážky. Vzdialenosť merajú tak, že počítajú čas, za ktorý sa im vyslané ultrazvukové vlny vrátia. Nameraná hodnota sa zobrazuje na displeji na palubnej doske. Parkovacie senzory má už skoro každé auto, lebo je to pohodlný a lacný asistenčný systém.

## Parkovacia kamera

Parkovacia kamera je rozšírenie k parkovacím senzorom. Kamera ukrytá buď vpredu alebo vzadu sleduje priestor pred alebo za autom a obraz premieta na displej na palubnej doske. Keď máme zaradenú spiatočku, tak sa zobrazuje obraz zo zadnej kamery a keď máme zaradené stupne 1 a viac a ideme 15 a menej km/h tak sa zobrazuje obraz z prednej kamery.

## Parkovací asistent

Parkovací asistent pomáha vodičovi parkovať. Sám krúti volantom a tak vodič musí len pridávať plyn a brzdiť. Tento asistenčný systém znižuje stres, zvyšuje komfort parkovania a pritom celý proces trvá iba 15 sekúnd. (22)

## Asistent rozjazdu do kopca

Asistent rozjazdu do kopca zabráni neželanému spätnému chodu automobilu na svahu. Tento asistenčný systém pôsobí v svahu s minimálnym sklonom 3 percentá. (23)

## Asistent zjazdu z kopca

Asistent zjazdu z kopca sám pribrzďuje kolesá keď ideme z kopca, takže vodič nemusí brzdiť ale len točiť volantom.

## Kontrola stavu pneumatík

Kontrola stavu pneumatík podporuje vodiča tým, že kontroluje tlak v pneumatikách. Vďaka neustálej kontrole zabezpečuje vysokú bezpečnosť posádky. Správnym tlakom v pneumatikách sa zvyšuje ich životnosť a znižuje spotreba paliva.

Sú dva spôsoby, ako kontrolovať stav pneumatík: priamy a nepriamy. Priamy využíva tlakový senzor priamo v pneumatike, a nepriamy pracuje prostredníctvom snímačov otáčok kolies. Pri znížení tlaku v pneumatike sa znižuje polomer kolesa a koleso sa pri rovnakej rýchlosti auta točí rýchlejšie.

Podľa spoločnosti Bridgestone 65 percent vodičov jazdí s podhustenými pneumatikami. Podhustené pneumatiky majú viacero nevýhod: pri poklese tlaku v pneumatike o 1 bar sa efektívnosť stabilizačného systému zníži o tretinu, spotreba sa zvýši o 0,3 l na 100 km, mäkká pneumatika taktiež predlžuje brzdnú dráhu, o 20 percent rýchlejšie sa opotrebúva pneumatika.

Ak by sa v Európskej únii zaviedlo povinné monitorovanie tlaku v pneumatikách, tak sa za jeden rok podarí ušetriť viac ako 1 miliardu litrov nafty alebo benzínu, čo znamená zníženie produkcie skleníkového CO2 o 4,8 milióna ton (24).

## Adaptívne predné svetlá – Dynamic light assist

Tento asistent spôsobí to, že po naštartovaní motora sa zapnú denné svetlá. Keď bude nižšia intenzita svetla, tak sa zapnú stretávacie svetlá. Pri jazde mimo obce sa automaticky zapnú diaľkové svetlá. Na základe kamery sa prispôsobuje tvar svetelného kužeľa podľa aktuálnej premávky (25).

## Dažďový senzor

Dažďový senzor pomocou fotodiódy a infračervených diód sleduje, či je na čelnom skle voda a ak áno, zapne stierače. Od čelného skla sa svetlo odráža a dopadá do fotodiódy, ktorá ho zachytí. Čím viac je na čelnom skle vody, tým menej svetla sa vráti do fotodiódy. Cez páčky stieračov sa dá nastaviť citlivosť senzora (26).



Obrázok – dažďový senzor

# Vlastný prínos

Ako vlastný prínos som vytvoril robotické auto ovládané z tabletu alebo z počítača, pre ktoré programujem a skúšam rôzne asistenčné systémy, napr. city safety. Robota som vyrobil doma zo stavebnice, ktorá obsahovala kolesá, motory a podvozok.

Inšpiroval som sa mojou sestrou Máriou, ktorá mala minulú ročníkovú prácu o robotoch. Jej najdokonalejší robot sa dal ovládať cez diaľkový ovládač. Sestre ocino kúpil stavebnicu robota s tromi kolesami, pričom iba dve boli poháňané – každé vlastným motorom. Mne ocino kúpil stavebnicu podobného robota, ale môj je väčší a viac sa podobá na auto. Má 4 kolesá, pričom každé je poháňané vlastným motorom, aby vedelo lepšie zatáčať. Mne sa toto auto páči aj zato, že vie zdolať aj náročnejší terén.

## Stavba robota

V stavebnici som dostal 4 kolesá, 4 motory, podvozok, držiak na 4 baterky, súčiastky na pripojenie motorov k podvozku a skrutky a matky (obrázok 17).

|  |  |
| --- | --- |
| C:\Users\Janko\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\IMG_6417.jpg  Obrázok - stavebnica robota | C:\Users\Janko\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\IMG_6497.jpg  Obrázok - prvý krok: montáž motorov |

**1. krok:** Najprv som na motory priletoval kábliky a pripojil ich k podvozku (obr. 18).

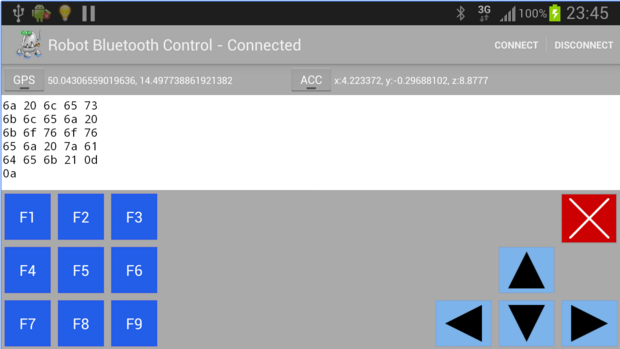
**2. krok**: Potom som na motory pridal kolesá. Na podvozok som pripojil držiak na baterky a vložil doňho baterky. Spravil som elektrický obvod, ktorý vypína a zapína motory pomocou vypínača. Obvod som zostavil na breadboarde.

Výsledkom je zostrojený robot, ktorý chodí dopredu. Ovláda sa vypínačom: v polohe „zapnuté“ ide, v polohe „vypnuté“ nejde.

|  |  |
| --- | --- |
| Obrázok - schéma základného obvodu pre robot | C:\Users\Janko\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\IMG_6505.jpg  Obrázok – prvý robot |

## Diaľkové ovládanie robota

Už dávnejšie ma zaujímajú autá na diaľkové ovládanie. Preto som sa rozhodol, že spravím robota ovládaného cez Bluetooth. V časopise Praktická elektronika sa o tomto písalo (27), ale robili to iba na mikrokontroléri PICAXE, ktorý sa programuje pomocou programovacieho jazyka BASIC. Ja som to však chcel robiť s Arduinom, lebo v ňom sa programuje v upravenom programovacom jazyku C++, ktorý je veľmi podobný Jave, o ktorej som mal ročníkovú prácu minulý rok. Tak som si na internete vyhľadal program, ktorý cez Bluetooth zapína a vypína LED (28). Upravil som ho na ovládanie viacerých diód[[1]](#footnote-1) a ovládanie rôznofarebných diód[[2]](#footnote-2). Nakoniec som tento program prepojil s programom na ovládanie robota cez diaľkový ovládač a mal som robota, ktorý bol ovládaný cez Bluetooth[[3]](#footnote-3). Na ovládanie robota som používal aplikáciu Robot Bluetooth Control, ktorú som stiahol z Google Play. Tento program vyzerá takto:



Obrázok - Robot Bluetooth Control

Ku každému tlačidlu treba zadať, že keď ho stlačíme, aké písmená má odoslať na pripojené zariadenie. Robot príjme písmeno z modulu Bluetooth a podľa toho, aké je to písmeno, ide buď dopredu, dozadu, doľava alebo doprava. Ak robot dostane písmeno A, tak nariadi kolesám, aby išli dopredu. Podobne, ak dostane písmená B, C, D alebo E, tak nariadi kolesám, aby buď išli doprava, doľava, dozadu alebo aby zastali. *Serial.println* je príkaz na odoslanie textu cez Bluetooth. Text uvidíme na obrazovke tabletu alebo PC.

void loop() {

 if(**Serial**.available()>0) {znak = **Serial**.read();}

 switch (znak){

   case 'A': dopredu();

**Serial**.println(" idem dopredu"); break;

   case 'B': doprava();

**Serial**.println(" idem doprava"); break;

   case 'C': dolava();

**Serial**.println(" idem dolava"); break;

   case 'D': dozadu();

**Serial**.println(" idem dozadu"); break;

   case 'E': vypniA(), vypniB();

**Serial**.println(" zastavam..."); break;

 }

 znak = 0;

}

Obrázok – Časť z programu, ktorý ovláda robota cez Bluetooth

## Asistenčné systémy

S týmto robotom som veľakrát narazil, preto som sa rozhodol spraviť asistenčný systém, ktorý pomáha vyhnúť sa nárazu. Na detekciu prekážky som použil ultrazvukový senzor vzdia­le­nosti. No predtým som si musel tento senzor vyskúšať. Otestoval som ho pomocou obvodu, kto­rý keď uvidí niečo bližšie ako 50 cm pred sebou, tak sa rozsvieti dióda na Arduine (obráz­ky 23 a 24). Program na Arduino som si stiahol z internetu (29). Cieľom otestovania bo­lo zistiť, aký presný je ultrazvukový senzor. Najprv som si odmeral vzdialenosť od senzora po stenu podľa ultrazvukového senzora a potom podľa dĺžkového meradla. Výsledok je, že ultra­­zvukový senzor je veľmi presný, má odchýlku max. 1 až 2 centimetre. (Jeden veľmi presný senzor som neskôr stratil v záhrade pri skúškach odolnosti robota pri prechode terénom.)

|  |  |
| --- | --- |
| C:\Users\Janko\Desktop\rp6 janko\foto\foto ultrazvukový deterktor\prvý pokus so ultrazvukovým detektorom\IMG_6601.JPG  Obrázok - obvod na skúšanie ultrazvukového senzora | C:\Users\Janko\Desktop\rp6 janko\foto\foto ultrazvukový deterktor\prvý pokus so ultrazvukovým detektorom\IMG_6587.JPG  Obrázok - skúšanie ultrazvukového senzora |

Keď som už ultrazvukový detektor vzdialenosti otestoval, tak som urobil aj môj prvý asistenčný systém – asistenčný systém upozornenia na prekážku. Tento systém upozorní správou na tablete, ak je robot vzdialený od prekážky 50 alebo menej centimetrov. Ak vodič nereaguje, keď je robot pri prekážke bližšie ako 15 cm, tak zastane[[4]](#footnote-4).

Potom som zostrojil asistenčný systém ktorý vie sám jazdiť. Keď uvidí prekážku bližšie ako 50 cm od seba, tak sa začne otáčať, pokým nenájde cestu, kde nie je prekážka bližšie ako 50 cm, a takto to pokračuje. Pri tomto asistenčnom systéme sa dá nastaviť, do akej strany má zabáčať, keď uvidí prekážku.

Ďalej som urobil „tempomat“, na ktorom sa dá nastaviť 8 rýchlostí motora.

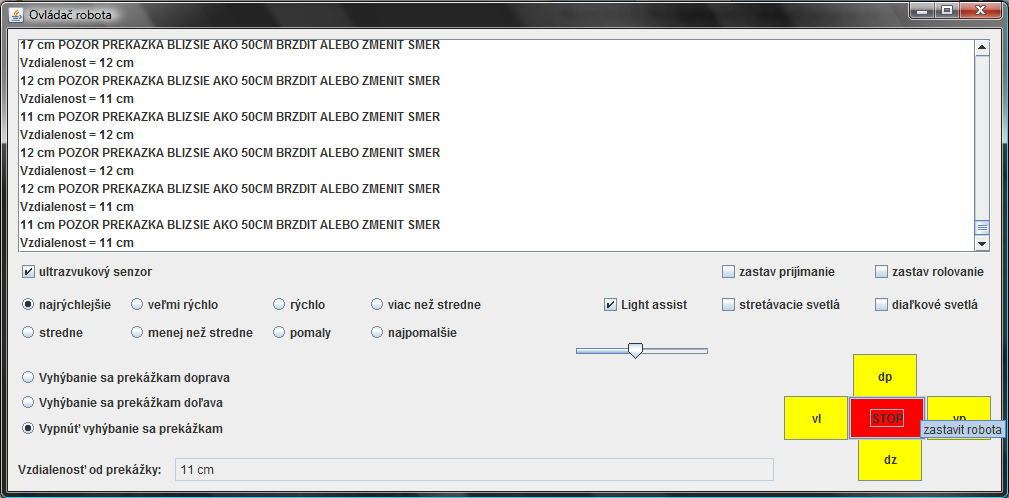
|  |  |
| --- | --- |
| Rýchlosť | Výkon (0- 255) |
| najrýchlejšie | 255 |
| veľmi rýchlo | 225 |
| rýchlo | 200 |
| viac než stredne rýchlo | 175 |
| stredne rýchlo | 150 |
| menej než stredne rýchlo | 125 |
| pomaly | 100 |
| najpomalšie | 75 |

Výkon motora sa reguluje pomocou špeciálnych pinov PWM (pulse-width modulation – pulzne šírková modulácia). PWM je rýchle striedanie 0 a 5 voltov, aby sa dosiahol taký istý výkon, aký by bol pri menšom napätí (30). Číslo 0-255 znamená, koľko času bude na výstupe zapnutých 5 voltov. Čím väčšie je číslo, tým viac času je na výstupe 5 voltov. 0 znamená, že je na výstupe stále 0, a 255 znamená, že je na výstupe stále 5 voltov.

Keď som robil tempomat, zistil som, že mám málo tlačidiel na nastavovanie rýchlostí pre tempomat. V Robot Bluetooth Control sa totiž nedajú pridávať tlačidlá. Tlačidiel na asistenčné systémy je tam 9 a ja som potreboval cca 13.

Preto som vytvoril vlastný program na počítači napísaný v programovacom jazyku Java, cez ktorý sa dá robot aj asistenčné systémy ovládať. Robil som to s pomocou knižnice *Bluecove*, ktorá slúži na pripojenie Javy na Bluetooth. Fungovala mi len na 32-bitových operačných systémoch.

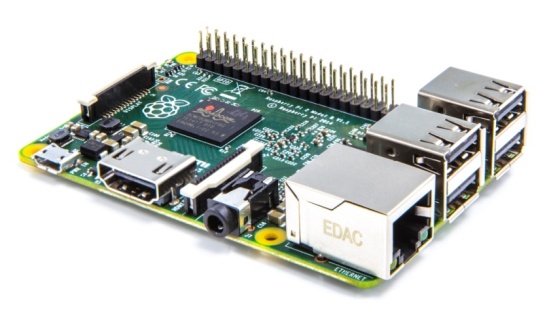
Tu môžeme vidieť okno tohto programu.



Obrázok - okno ovládacieho programu

Program je v prílohe a na stránke <https://github.com/janmederly/rp6/tree/master/java/Robot>.

Potom som išiel pridať do môjho robota kameru napojenú na minipočítač Raspberry Pi, ktorý by odosielal obraz z kamery pomocou WiFi do notebooku. Raspberry Pi je séria minipočítačov so špeciálne upraveným operačným systémom Linux, dá sa na ňom však použiť aj Windows 10 IoT Core.



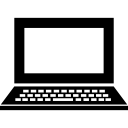
Obrázok - Raspberry Pi

Najprv som si myslel, že to bude jednoduché, ale zistil som, že práve opak je pravdou. Prvý problém bol, že kamera dodávaná priamo k minipočítaču Raspberry Pi obraz premietala rovno na obrazovku, ktorú som zo začiatku naňho pripojil cez port HDMI. Obraz nešiel cez procesor, čiže sa nedal posielať priamo cez WiFi. S vedúcim sme to vyskúšali spraviť pomocou streamovania podľa návodu z internetu (31). Streamovanie sa však aj tak nedarilo. Na napájanie Raspberry Pi sme totiž používali batériu slúžiacu na nabíjanie mobilov (power bank). Batéria nevedela dodať dostatok elektriny pre Raspberry Pi. Potom som sa Raspberry Pi pokúsil pripojiť na baterky, ktoré poháňajú motory. Pri tejto príležitosti som rozobral celého robota a vyrobil špeciálny „napájací systém“ podľa návrhu vedúceho.

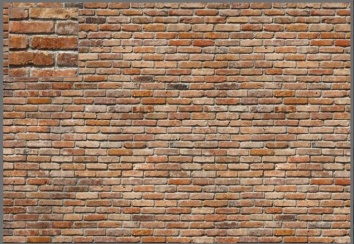
Nakoniec som robotovi spravil svetlá: diaľkové, stretávacie, denné, zadné, brzdové, smerovky a spiatočku. Robot okrem toho obsahuje aj „Light assist“, ktorý podľa toho, či je tma alebo svetlo, zapína alebo vypína svetlá.

## Zhrnutie funkcií robota

Tento robot sa dokáže pohybovať, a to viacerými rýchlosťami a smermi (dopredu, dozadu, vpravo, vľavo). Vieme ho ovládať cez tablet alebo počítač. Dokáže informovať vodiča o vzdialenosti od prekážky, zastaviť sám pred prekážkou a vyhnúť sa prekážke. Má na sebe namontovanú kameru, vie odosielať signál z kamery do počítača cez WiFi a vie ukladať obraz z kamery do svojho úložiska alebo do úložiska počítača.



Bluetooth + WiFi



Ultrazvukový senzor vzdialenosti

kamera

Obrázok - scbéma prostredia robota

## Súčasti systému

Tento robot obsahuje nasledujúce súčasti:

1. skladačka robota (podvozok, kolesá, motory),
2. Arduino Uno (neskôr Mega) a ovládač motorov,
3. batérie LiPo,
4. obvod na komunikáciu cez Bluetooth (HC-06),
5. ultrazvukový senzor vzdialenosti (HC SR-04),
6. napájací systém,
7. Raspberry Pi verzia 2 model B,
8. kamera RaspiCam (originálne príslušenstvo k Raspberry Pi),
9. WiFi USB modul,
10. LED diódy (svetlá),
11. fotorezistor,
12. káble a pomocné súčiastky.

Arduino

Ovládač motorov

Motory

UZV senzor

Bluetooth

Raspberry Pi

Kamera

Svetlá

Fotorezistor

Baterky

Napájací systém

WiFi

Obrázok - schéma súčastí robota

**Arduino Uno (Mega)** je ako keby mozog celého robota. Je to obvod, na ktorom sa nachádza mikrokontrolér Atmel Atmega328 (resp. Atmega2560) a pomocné súčiastky. Mikrokontrolér je jednočipový počítač. Obsahuje procesor, pamäť RAM, FLASH a EEPROM, časovače, vstupno-výstupné zariadenia: kontrolér sériovej komunikácie, A/D a D/A prevodníky, vstupno-výstupné porty (piny). (32) Riadiaci program pre mikrokontrolér je v prílohe a na stránke <https://github.com/janmederly/rp6/tree/master/arduino/p7_robot_so_svetlami>.

**Ovládač motorov** má súčiastku, ktorá pomocou malého prúdu z Arduina (do 20 mA) vie vyvolať väčší prúd, ktorý dokáže poháňať motory (do 2000 mA).

Dve nabíjateľné **batérie LiPo**, s kapacitou 6600 mAh každá, dodávajú dosť veľký prúd na to, aby sa motory vedeli pohnúť a pomocou veľkej kapacity zabezpečia dlhší chod motorov. Najprv som skúšal 12 bateriek AA, ale rozhodol som sa, že radšej dám batérie LiPo. Majú väčšiu kapacitu a je to lacnejšie, lebo nemusíme neustále kupovať nové baterky. Batérie LiPo sú dve, lebo minimálne napätie pre Arduino je 5 V a pre ovládač motorov 6 V. Jedna maximálne nabitá batéria LiPo má 4,12 V (zistil som to pokusom).

**Obvod na komunikáciu cez Bluetooth (HC-06)** slúži na komunikáciu s tabletom alebo počítačom alebo mobilom. Bluetooth je otvorený štandard pre bezdrôtovú komunikáciu elektronických zariadení.

**Ultrazvukový senzor vzdialenosti (HC SR-04)** pomocou ultrazvukových vĺn meria vzdialenosť. Senzor vyšle ultrazvukové vlny a počíta, kedy sa vráti ich ozvena. Vzdialenosť (d) sa meria podľa toho, koľko času (t) ubehlo odkedy senzor odoslal vlny až po prijatie ozveny. Vzdialenosť d = (t × 340 m/s) : 2. (340 m/s je rýchlosť zvuku vo vzduchu.)

**Napájací systém** slúži na napájanie viacerých zariadení z jednej sady bateriek. Pre Raspberry Pi upravuje napätie presne na 5 voltov. Pre Arduino obsahuje ochranu pred vysokým prúdom pri zapnutí motorov. Vysoké prúdy vznikajú tak, že motory potrebujú veľký prúd, aby sa rozbehli, ale baterky s ochranou proti skratu nedovolia dodať taký veľký prúd. Prejavovalo sa to tak, že pri zapnutí motorov sa robot niekedy vypol.

**Raspberry Pi verzia 2 model B** slúži na spracovanie obrazu z kamery a poslanie ho do počítača cez WiFi modul.

**Fotorezistor** mení svoj odpor podľa toho, koľko svetla naňho dopadá.

Teraz si opíšeme schémy zapojení robota.

Na obrázku 29 môžeme vidieť riadiacu časť robota – Arduino, ovládač motorov (AMS00298PM), ultrazvukový senzor (HC SR04), Bluetooth vysielač a prijímač (HC 06).

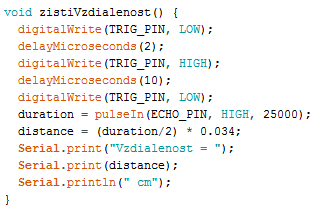
Arduino je spojené s ovládačom motorov pomocou pinov D8-D13. Sú to digitálne piny, to znamená, že sa nich dá nastaviť buď napätie 0 voltov alebo 5 voltov. Ovládač motorov vie ovládať dva motory. Keď dáme na pin D8 nulu a na pin D11 jednotku, prvý motor pôjde v smere hodinových ručičiek. Keď dáme piny naopak, motor pôjde proti smeru hodinových ručičiek. Pin D9 slúži na nastavenie rýchlosti prvého motora. Podobne slúžia piny D12, D13 a D10 na ovládanie smeru a rýchlosti druhého motora (33). Ja, keďže mám štyri motory, som dva motory na pravej strane zapojil na ovládač ako jeden motor. Podobne som spravil s dvoma motormi na ľavej strane.

Ultrazvukový senzor HC SR04 má 4 piny: Vcc, GND, Trig a Echo. Na Vcc dáme +5 voltov, na GND zem. Na pin Trig posielame signál, aby sa ultrazvukový senzor rozbehol. Na Echo pine ultrazvukový senzor dáva vlnu, ktorá trvá tak dlho, aký čas bol medzi odoslaním ultrazvuku a prijatím ozveny. Trig pin pripojíme na pin D2 a Echo pin na pin D3.

Obrázok - schéma robota bez svetiel



Odošleme krátky signál, aby senzor odoslal ultrazvukové vlny.



Do premennej distance si vypočítame vzdialenosť od prekážky.

Pomocou funkcie pulseIn zistíme čas ktorý zvuk potrebuje, aby sa dostal k prekážke a naspäť

Obrázok - narábanie s ultrazvukovým senzorom

Na obrázku 30 môžeme vidieť časť z programu zastavenia pred prekážkou. Najprv do pinu Trig pošleme krátky signál (10 µs), aby ultrazvukový senzor odoslal ultrazvukové vlny. Funkcia *pulseIn* čaká, pokým na pine, ktorý si zvolíme nebude HIGH. Keď bude už na pine HIGH, začne počítať, pokým nebude znova na pine LOW. Výsledok odovzdáva v mikrosekundách, alebo vráti 0, keď žiaden kompletný signál počas timeoutu nedostane. (34) Ja som zadal timeout 25000 mikrosekúnd.

Potom vzdialenosť vypočítame takto: vzdialenosť = čas × rýchlosť. Ako čas zoberieme premennú duration/2, lebo duration je čas, ktorý potrebuje zvuk na to, aby došiel k prekážke a naspäť. My však chceme zistiť iba vzdialenosť k prekážke, preto duration vydelíme dvomi. Ako rýchlosť si zoberieme hodnotu 0,034 cm/µs, lebo rýchlosť zvuku je 340 m/s, ale my potrebujeme vedieť rýchlosť zvuku v centimetroch za mikrosekundu. 340 m/s = 34000 cm/s = 0,034 cm/µs.

Bluetooth modul HC 06 má 4 piny: Vcc, GND, TXD, RXD. Na Vcc pripojíme +5 voltov, na GND zem. TXD slúži na odosielanie informácií z Bluetoothu do Arduina. RXD slúži na odosielanie informácií z Arduina cez Bluetooth. TXD som zapojil na Arduinový pin D0-RX. Ten slúži na prijímanie informácií do Arduina. RXD som zapojil na Arduino pin D1-TX, ktorý slúži na odosielanie informácií z Arduina. Logic level convertor je tam preto, lebo Bluetooth modul potrebuje prijímať signál pri napätí 3,3 volta a Arduino pri 5 voltov.

Na obrázku 31 vidíme schému napájacieho systému. Vyhla­dzo­vač slúži na vyhladenie silného prúdu, ktorý si žiadajú motory, keď sa zapínajú. Pozostáva z cievky L1 a diódy D1. Cievka slúži na vyhladenie veľkého prúdu, ktorý si žiadajú motory pri zapnutí. Baterky s ochranou proti veľkému odberu nedovolia vydať taký veľký prúd a tak sa vypnú. Dióda D1 slúži na vyrovnanie opačného napätia, ktoré sa indukuje na cievke L1 po vypnutí motorov. Regulátor napätia slúži na to, že Raspberry Pi vyžaduje presne 5 voltov. Batérie dávajú okolo 8 voltov. Voltmeter sa používa na to, aby som vedel, aké napätie je na baterkách, tzn. ako sú baterky nabité. Dióda D2 slúži na to, aby som vedel, či je napätie na USB porte (USB port slúži na napájanie Raspberry Pi).

Obrázok - schéma napájacieho systému

Na obrázku 32 vidíme schému zapojenia fotorezistora a svetiel. Moje auto má denné, stretávacie, diaľkové svetlá, smerovky, zadné, brzdové svetlá a spiatočku. Svetlá sú urobené z LED diód a rezistorov. Svetlá sú ovládané Arduinom. Sú pripojené na digitálne piny. Stretávacie svetlá sú pripojené na PWM piny, aby keď do nich pustíme menší výkon, mohli svietiť ako denné svetlá. Ako smerovky som použil blikajúce žlté diódy, aby som to nemusel programovať na Arduine. Na použitie svetiel som musel vymeniť Arduino UNO za Arduino Mega, aby som mal dosť pinov na pripojenie diód.

Spravil som špeciálny asistenčný systém „Light assist“, ktorý podľa toho, či je tma alebo nie, zasvieti diaľkové alebo denné svetlá. Na zistenie, či je tma alebo nie, používam fotorezistor R14. Na to, aby som zistil, aký má fotorezistor práve odpor, som použil Ohmov zákon:



Obrázok - schéma svetiel a systému Light assist

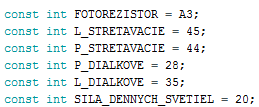
K fotorezistoru R14 som zapojil druhý rezistor R15 s odporom R2 = 2,2 MΩ. Pomocou analógového vstupného pinu A3 meriam napätie U na fotorezistore. Napätie U2 na známom odpore R15 si vypočítam tak, že od 5 voltov odpočítam namerané napätie na fotorezistore. Potom si vypočítam prúd, ktorý preteká na známom rezistore R15, lebo taký istý prúd bude pretekať aj na fotorezistore. Prúd si vypočítam pomocou Ohmovho zákona:

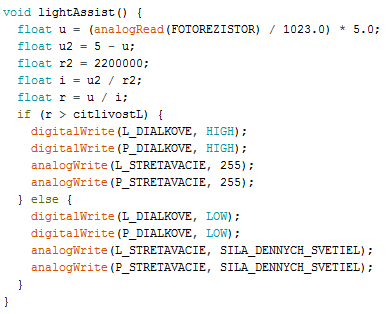
I = =

Teraz, keď vieme prúd, si pomocou Ohmovho zákona vypočítame odpor fotorezistora R:

R =

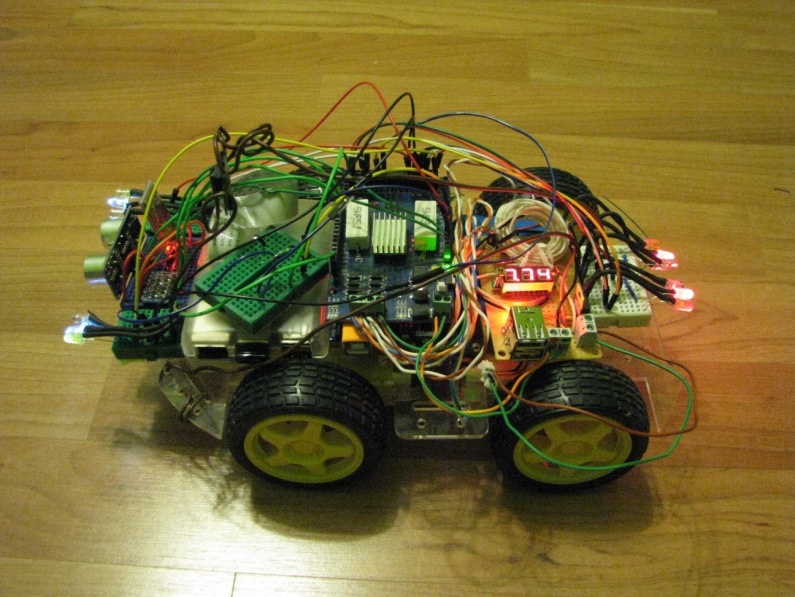
Takto to vyzerá naprogramované:





Obrázok - program pre Light assist

Takto vyzerá dokončený robot.



Obrázok - hotový robot

# Záver

V tejto práci som sa venoval asistenčným systémom v automobiloch. Teoretická časť obsahuje opisy 20 vybraných asistenčných systémov. Ako vlastný prínos som urobil robota, na ktorom som skúšal viaceré asistenčné systémy: asistent zabrzdenia pred prekážkou, asistent vyhýbania sa prekážke, tempomat, automatické zapínanie svetiel a kameru.

Asistent zabrzdenia pred prekážkou a asistent vyhýbania sa prekážke spoľahlivo fungujú na rovnom povrchu s veľkými prekážkami, ako je napríklad stena. Naopak, nefungujú pri hrboľatom teréne (napr. v záhrade), prípadne na miernom kopci. Nevšimnú si prekážku, ktorá je príliš úzka, ako je napríklad noha stoličky.

Tempomat udržiava nastavený výkon motorov. Nie je to „pravý“ tempomat, pretože sa nesnaží dodržať nastavenú rýchlosť, lebo nevie, aká je reálna rýchlosť robota.

Automatické zapínanie svetiel funguje bez chyby.

Kamera funguje, ale niekedy sa na počítači zobrazuje obraz s veľkým zdržaním. Na opravení tejto chyby ešte pracujem.

V budúcnosti plánujem na robota pridať ešte ďalšie možnosti, napríklad kompas alebo akcelerometer.

# Summary

This work is about driver assistance systems in cars. The theoretical part contains descriptions of 20 selected driver assistance systems. As own contribution I created a robot on which I implemented a number of assistance systems: stopping before obstacles, avoiding obstacles, cruise control, automatic switching of lights, and a camera. In future, I plan to add to my robot more features, for example a compass or an accelerometer.

# Použité zdroje

1. **Struth, W.** Bezpečnosť, komfort a agilita – asistenčné systémy pre vodiča od firmy Bosch. [Online] 6 2009. [Dátum: 27. 4. 2016.] http://press.bosch.sk/press/upload/043ks-d\_\_MPK09\_\_driver\_assistance\_systems\_sk.pdf.

2. Úmrtí při dopravních nehodách v Evropě. *Wikipédia.* [Online] [Dátum: 27. 4. 2016.] https://cs.wikipedia.org/wiki/%C3%9Amrt%C3%AD\_p%C5%99i\_dopravn%C3%ADch\_nehod%C3%A1ch\_v\_Evrop%C4%9B.

3. **BECEP.** Nárast nehodovosti. [Online] 28. 10. 2014. [Dátum: 27. 4. 2016.] http://www.becep.sk/aktuality/117/narast-nehodovosti.

4. Bezpečnostný vzduchový vankúš. *Wikipédia.* [Online] [Dátum: 27. 4. 2016.] https://sk.wikipedia.org/wiki/Bezpe%C4%8Dnostn%C3%BD\_vzduchov%C3%BD\_vank%C3%BA%C5%A1.

5. ABS (vozidlo). *Wikipédia.* [Online] [Dátum: 27. 4. 2016.] https://sk.wikipedia.org/wiki/ABS\_(vozidlo).

6. **Sajdl, J.** ABS (Anti-lock Braking System). *autolexicon.net.* [Online] 28. 3. 2011. [Dátum: 27. 4. 2016.] http://www.autolexicon.net/cs/articles/abs-anti-lock-braking-system/.

7. **Klus, J.** Snímače. [Online] [Dátum: 27. 4. 2016.] http://files.jozefklus.webnode.sk/200000173-70190720f7/2%20-%20Elektrick%C3%BD%20rozvod%20-%203.cast%20-%20snima%C4%8De.pdf.

8. **Volkswagen.** Technický lexikón: Antiblokovací systém (ABS). [Online] [Dátum: 27. 4. 2016.] http://w3.vw.sk/\_cms/lexikon/antiblokovaci\_system\_abs.html.

9. Regulácia preklzovania. *Wikipédia.* [Online] [Dátum: 27. 4. 2016.] https://sk.wikipedia.org/wiki/Regul%C3%A1cia\_preklzovania.

10. **Škoda Auto.** Protipreklzový systém - ASR. [Online] [Dátum: 27. 4. 2016.] http://www.skoda-auto.sk/models/hotspotdetail?HotspotName=asr&Page=technology&WebID=031bd6df-c1f8-4ed4-b67f-5fcf6c1dd1b9.

11. Elektronický stabilizačný systém. *Wikipédia.* [Online] [Dátum: 27. 4. 2016.] https://sk.wikipedia.org/wiki/Elektronick%C3%BD\_stabiliza%C4%8Dn%C3%BD\_syst%C3%A9m.

12. **BECEP.** ESP je jednoznačný pomocník! [Online] 23. 3. 2013. [Dátum: 27. 4. 2016.] http://www.becep.sk/vodici/37/esp-je-jednoznacny-pomocnik.

13. **Sajdl, J.** ESP (Electronic Stability Programme). *autolexicon.net.* [Online] 5. 4. 2011. [Dátum: 27. 4. 2016.] http://www.autolexicon.net/cs/articles/esp-electronic-stability-programme/.

14. **Volkswagen.** Technický lexikón: Elektronický stabilizačný program (ESP). [Online] [Dátum: 27. 4. 2016.] http://w3.vw.sk/\_cms/lexikon/elektronicky\_stabiliza\_ny\_program\_esp.html.

15. **Andrejčák, T.** ESP je od novembra povinné. Autá sa bez neho predávať nesmú. *auto.pravda.sk.* [Online] 4. 11. 2014. [Dátum: 27. 4. 2016.] http://auto.pravda.sk/novinky/clanok/335187-esp-je-od-novembra-povinne-v-europe-sa-auta-bez-neho-predavat-nesmu/.

16. **Garaj, F.** Renault Laguna 4Control skúša oklamať fyziku. [Online] 25. 10. 2011. [Dátum: 27. 4. 2016.] http://auto.sme.sk/c/6111342/renault-laguna-4control-skusa-oklamat-fyziku.html.

17. **Volkswagen.** Brzdový asistent - v núdzi zastavíte rýchlejšie. [Online] [Dátum: 27. 4. 2016.] http://www.vw.sk/svet\_volkswagen/inovacie\_technika/asisten\_ne\_systemy/brzdovy\_asistent.

18. **Sajdl, J.** Multikolízna brzda. *autolexicon.net.* [Online] 23. 5. 2015. [Dátum: 27. 4. 2016.] http://www.autolexicon.net/sk/articles/multikolizni-brzda/.

19. **Volkswagen.** Side Assist "Plus". [Online] [Dátum: 27. 4. 2016.] http://www.vw.sk/modely/volkswagen\_cc/vlastnosti/81048\_side\_assist\_plus.

20. **Volkswagen.** Technický lexikón: Lane Assist. [Online] [Dátum: 27. 4. 2016.] http://w3.vw.sk/\_cms/lexikon/lane\_assist.html.

21. **Volkswagen.** Sign Assist. [Online] [Dátum: 27. 4. 2016.] http://www.volkswagen.co.uk/technology/driving/sign-assist.

22. **Volkswagen.** Technický lexikón: Park Assist. [Online] [Dátum: 27. 4. 2016.] http://w3.vw.sk/\_cms/lexikon/park\_assist.html.

23. **Citroen.** Asistent rozjazdu do kopca. [Online] [Dátum: 27. 4. 2016.] http://www.citroen.sk/technologie/asistent-rozjazdu-do-kopca.html.

24. **Andrejčák, T.** Monitorovanie tlaku pneumatík bude povinné. Aj na STK. *auto.pravda.sk.* [Online] 24. 10. 2014. [Dátum: 27. 4. 2016.] http://auto.pravda.sk/poradna/clanok/334176-monitorovanie-tlaku-pneumatik-bude-povinne-aj-na-stk/.

25. **Volkswagen.** Technický lexikón: Dynamic Light Assist. [Online] [Dátum: 27. 4. 2016.] http://w3.vw.sk/\_cms/lexikon/dynamic\_light\_assist.html.

26. **Volkswagen.** Technický lexikón: Dažďový senzor. [Online] [Dátum: 27. 4. 2016.] http://w3.vw.sk/\_cms/lexikon/da\_ovy\_senzor.html.

27. *Využití Bluetooth z mobilu, tabletu nebo PC k ovládání.* **Černý, M.** 2, 2015, Praktická elektronika, Zv. XX (LXIV), s. 20-23. ISSN 1804-7173.

28. **yhtomitsy.** Add bluetooth to your Arduino project - Arduino+HC-06. *Instructables.* [Online] [Dátum: 28. 4. 2016.] http://www.instructables.com/id/Add-bluetooth-to-your-Arduino-project-ArduinoHC-06/.

29. **jsvester.** Simple Arduino and HC-SR04 Example. *Instructables.* [Online] [Dátum: 27. 4. 2016.] http://www.instructables.com/id/Simple-Arduino-and-HC-SR04-Example/.

30. **Božík, M.** *Ako naprogramovať Arduino bez predchádzajúcich znalostí.* 2013.

31. **Bears, Z.** http://zacharybears.com/low-latency-raspberry-pi-video-streaming/. [Online] [Dátum: 27. 4. 2016.] http://zacharybears.com/low-latency-raspberry-pi-video-streaming/.

32. Mikrokontrolér. *Wikipédia.* [Online] [Dátum: 27. 4. 2016.] https://sk.wikipedia.org/wiki/Mikrokontrol%C3%A9r.

33. **Elecrow.** Motor&Stepper Shield. [Online] [Dátum: 27. 4. 2016.] http://www.elecrow.com/wiki/index.php?title=Motor%26Stepper\_Shield.

34. **Arduino.** Arduino Language Reference. [Online] 27. 4. 2016. https://www.arduino.cc/en/Reference/HomePage.

# Príloha

## Program pre Arduino (na robotovi)

const int TRIG\_PIN = 2;

const int ECHO\_PIN = 3;

const int SMER\_A1 = 8;

const int SMER\_A2 = 11;

const int RYCHLOST\_A = 9;

const int SMER\_B1 = 12;

const int SMER\_B2 = 13;

const int RYCHLOST\_B = 10;

const int PL\_SMEROVKA = 29;

const int PP\_SMEROVKA = 47;

const int ZL\_SMEROVKA = 27;

const int ZP\_SMEROVKA = 32;

const int L\_STRETAVACIE = 45;

const int P\_STRETAVACIE = 44;

const int P\_DIALKOVE = 28;

const int L\_DIALKOVE = 35;

const int P\_ZADNE = 23;

const int L\_ZADNE = 50;

const int P\_BRZDNE = 22;

const int L\_BRZDNE = 51;

const int SPIATOCKA = 49;

const int SILA\_DENNYCH\_SVETIEL = 20;

const int FOTOREZISTOR = A3;

void setup() {

**Serial**.begin (9600);

 pinMode(TRIG\_PIN, OUTPUT);

 pinMode(ECHO\_PIN, INPUT);

 pinMode(SMER\_A1, OUTPUT);

 pinMode(SMER\_A2, OUTPUT);

 pinMode(RYCHLOST\_A, OUTPUT);

 pinMode(SMER\_B1, OUTPUT);

 pinMode(SMER\_B2, OUTPUT);

 pinMode(RYCHLOST\_B, OUTPUT);

 pinMode(PP\_SMEROVKA, OUTPUT);

 pinMode(PL\_SMEROVKA, INPUT);

 pinMode(ZP\_SMEROVKA, OUTPUT);

 pinMode(ZL\_SMEROVKA, OUTPUT);

 pinMode(L\_STRETAVACIE, OUTPUT);

 pinMode(P\_STRETAVACIE, OUTPUT);

 pinMode(P\_DIALKOVE, OUTPUT);

 pinMode(L\_DIALKOVE, OUTPUT);

 pinMode(P\_ZADNE, OUTPUT);

 pinMode(L\_ZADNE, OUTPUT);

 pinMode(P\_BRZDNE, OUTPUT);

 pinMode(L\_BRZDNE, OUTPUT);

 pinMode(SPIATOCKA, OUTPUT);

 svetla();

}

const int DOPREDU = 1;

const int DOZADU = 2;

const int DOPRAVA = 3;

const int DOLAVA = 4;

const int STAT = 0;

long citlivostL = 0;

boolean jeZapnutyL = false;

long vbs = 0;

int smer = STAT;

boolean zapnutyAsistentNarazu = 1;

int jeZapnutyACC = 0;

int ktoraRychlost = 0;

long distance;

void loop(){

 prikazZTabletu();

 if (zapnutyAsistentNarazu) {

   testVzdialenost();

 }

 if (jeZapnutyL) {

   lightAssist();

 }

 if (vbs > 0 && millis() >= vbs){

   digitalWrite(P\_BRZDNE, LOW);

   digitalWrite(L\_BRZDNE, LOW);

 }

 diagnostika();

 delay (200);

}

void diagnostika() {

**Serial**.print("[");

**Serial**.print("D=");

**Serial**.print(distance);

**Serial**.print(" T=");

**Serial**.print(ktoraRychlost);

**Serial**.print(" S=");

**Serial**.print(smer);

**Serial**.print(" L");

**Serial**.print(jeZapnutyL);

**Serial**.print(",");

**Serial**.print(citlivostL);

**Serial**.print(" AN=");

**Serial**.print(zapnutyAsistentNarazu);

**Serial**.print(" O=");

**Serial**.print(jeZapnutyACC);

 //[D=34 T1 S1 L+8 N+ OP]

**Serial**.println("]");

}

void lightAssist() {

 float u = (analogRead(FOTOREZISTOR) / 1023.0) \* 5.0;

 float u2 = 5 - u;

 float r2 = 2200000;

 float i = u2 / r2;

 float r = u / i;

 if (r > citlivostL) {

   digitalWrite(L\_DIALKOVE, HIGH);

   digitalWrite(P\_DIALKOVE, HIGH);

   analogWrite(L\_STRETAVACIE, 255);

   analogWrite(P\_STRETAVACIE, 255);

 } else {

   digitalWrite(L\_DIALKOVE, LOW);

   digitalWrite(P\_DIALKOVE, LOW);

   analogWrite(L\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

   analogWrite(P\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

 }

}

void prikazZTabletu(){

 int znak = 0;

 if(**Serial**.available()>0) {znak = **Serial**.read();}

 switch (znak) {

   case 'A': dopredu();

**Serial**.println(" idem dopredu");

     smer = DOPREDU;

     break;

   case 'C': doprava();

**Serial**.println(" idem doprava");

     smer = DOPRAVA;

     break;

   case 'B': dolava();

**Serial**.println(" idem dolava");

     smer = DOLAVA;

     break;

   case 'D': dozadu();

**Serial**.println(" idem dozadu");

     smer = DOZADU;

     break;

   case 'E': zastav();

**Serial**.println(" zastavam...");

     smer = STAT;

     break;

   case 'F': zapnutyAsistentNarazu = true;

**Serial**.println("asistent narazu zapnuty");

     break;

   case 'G': zapnutyAsistentNarazu = false;

**Serial**.println("asistent narazu vypnuty");

     break;

   case 'H': ktoraRychlost = 1;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"najrychlejsie\"");

     break;

   case 'I': ktoraRychlost = 2;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"velmi rychlo\"");

     break;

   case 'J': ktoraRychlost = 3;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"rychlo\"");

     break;

   case 'K': ktoraRychlost = 4;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"viac nez stredne\"");

     break;

   case 'L': ktoraRychlost = 5;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"stredne\"");

     break;

   case 'M': ktoraRychlost = 6;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"menej nez stredne\"");

     break;

   case 'N': ktoraRychlost = 7;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"pomaly\"");

     break;

   case 'O': ktoraRychlost = 8;

     if (smer != STAT) {

       nastavRychlostMotorov(0);

     }

**Serial**.println("nastaveny tempomat \"najpomalsie\"");

     break;

   case 'Q':

     jeZapnutyACC = DOPRAVA;

**Serial**.println("nastavene vyhybanie sa prekazkam doprava");

     break;

   case 'P':

     jeZapnutyACC = DOLAVA;

**Serial**.println("nastavene vyhybanie sa prekazkam dolava");

     break;

   case 'R': jeZapnutyACC = 0;

**Serial**.println("vypnute vyhybanie sa prekazkam");

     break;

   case 'S':

     analogWrite(L\_STRETAVACIE, 255);

     analogWrite(P\_STRETAVACIE, 255);

     jeZapnutyL = false;

     break;

   case 'T': analogWrite(L\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

     analogWrite(P\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

     break;

   case 'U': analogWrite(L\_STRETAVACIE, 255);

     analogWrite(P\_STRETAVACIE, 255);

     digitalWrite (L\_DIALKOVE, HIGH);

     digitalWrite (P\_DIALKOVE, HIGH);

     jeZapnutyL = false;

     break;

   case 'V':

     digitalWrite (L\_DIALKOVE, LOW);

     digitalWrite (P\_DIALKOVE, LOW);

     break;

   case 'W':

     {

       jeZapnutyL = true;

       long cas = millis();

       while(**Serial**.available() <= 0 && millis()-cas < 1000) {

       }

       if (**Serial**.available() > 0) {

         int dalsi = **Serial**.read();

         citlivostL = (dalsi-'0'+1) \* 8000000;

       }

     }

     break;

   case 'X': jeZapnutyL = false;

     digitalWrite (L\_DIALKOVE, LOW);

     digitalWrite (P\_DIALKOVE, LOW);

     analogWrite(L\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

     analogWrite(P\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

     break;

 }

}

void testVzdialenost() {

 zistiVzdialenost();

 if (distance >= 400 || distance <= 0) {

**Serial**.println ("ste pridaleko alebo priblizko k prekazke");

 } else if (distance < 50) {

**Serial**.print(distance);

   if (jeZapnutyACC == DOPRAVA && smer == DOPREDU) {

     otacajSaKymJePrekazkaDoprava();

   } else if (jeZapnutyACC == DOLAVA && smer == DOPREDU) {

     otacajSaKymJePrekazkaDolava();

   } else {

**Serial**.println(" cm POZOR PREKAZKA BLIZSIE AKO 50 CM BRZDIT ALEBO ZMENIT SMER ");

     if (distance < 25 && smer == DOPREDU) {

        zastav();

     }

   }

 }

}

void svetla() {

  digitalWrite(P\_ZADNE, HIGH);

  digitalWrite(L\_ZADNE, HIGH);

  digitalWrite(P\_DIALKOVE, LOW);

  digitalWrite(L\_DIALKOVE, LOW);

  digitalWrite(PP\_SMEROVKA, LOW);

  digitalWrite(PL\_SMEROVKA, LOW);

  digitalWrite(ZP\_SMEROVKA, LOW);

  digitalWrite(ZL\_SMEROVKA, LOW);

  digitalWrite(P\_BRZDNE, LOW);

  digitalWrite(L\_BRZDNE, LOW);

  digitalWrite(SPIATOCKA, LOW);

  analogWrite(L\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

  analogWrite(P\_STRETAVACIE, SILA\_DENNYCH\_SVETIEL);

}

void dopredu() {

 smerA(1);

 smerB(1);

 nastavRychlostMotorov(255);

 digitalWrite(PP\_SMEROVKA, LOW);

 digitalWrite(PL\_SMEROVKA, LOW);

 digitalWrite(ZP\_SMEROVKA, LOW);

 digitalWrite(ZL\_SMEROVKA, LOW);

 digitalWrite(P\_BRZDNE, LOW);

 digitalWrite(L\_BRZDNE, LOW);

 digitalWrite(SPIATOCKA, LOW);

}

void dozadu() {

 smerA(0);

 smerB(0);

 nastavRychlostMotorov(255);

 digitalWrite(SPIATOCKA, HIGH);

 digitalWrite(PP\_SMEROVKA, LOW);

 digitalWrite(PL\_SMEROVKA, LOW);

 digitalWrite(ZP\_SMEROVKA, LOW);

 digitalWrite(ZL\_SMEROVKA, LOW);

 digitalWrite(P\_BRZDNE, LOW);

 digitalWrite(L\_BRZDNE, LOW);

}

void doprava() {

 smerA(1);

 smerB(0);

 nastavRychlostMotorov(150);

 digitalWrite(PP\_SMEROVKA, HIGH);

 digitalWrite(ZP\_SMEROVKA, HIGH);

 digitalWrite(PL\_SMEROVKA, LOW);

 digitalWrite(ZL\_SMEROVKA, LOW);

 digitalWrite(P\_BRZDNE, LOW);

 digitalWrite(L\_BRZDNE, LOW);

 digitalWrite(SPIATOCKA, LOW);

}

void dolava() {

 smerA(0);

 smerB(1);

 nastavRychlostMotorov(150);

 digitalWrite(PL\_SMEROVKA, HIGH);

 digitalWrite(ZL\_SMEROVKA, HIGH);

 digitalWrite(PP\_SMEROVKA, LOW);

 digitalWrite(ZP\_SMEROVKA, LOW);

 digitalWrite(P\_BRZDNE, LOW);

 digitalWrite(L\_BRZDNE, LOW);

 digitalWrite(SPIATOCKA, LOW);

}

void nastavRychlostMotorov(int rychlostBezTempomatu) {

 switch (ktoraRychlost) {

   case 1: rychlostA(255); rychlostB(255); break;

   case 2: rychlostA(225); rychlostB(225); break;

   case 3: rychlostA(200); rychlostB(200); break;

   case 4: rychlostA(175); rychlostB(175); break;

   case 5: rychlostA(150); rychlostB(150); break;

   case 6: rychlostA(125); rychlostB(125); break;

   case 7: rychlostA(100); rychlostB(100); break;

   case 8: rychlostA(75); rychlostB(75); break;

   case 0: rychlostA(rychlostBezTempomatu); rychlostB(rychlostBezTempomatu); break;

 }

}

void otacajSaKymJePrekazkaDoprava(){

 doprava();

 otacajSaKymJePrekazka();

}

void otacajSaKymJePrekazkaDolava(){

 dolava();

 otacajSaKymJePrekazka();

}

void otacajSaKymJePrekazka(){

 rychlostA(100);

 rychlostB(100);

 while (distance <= 50) {

   zistiVzdialenost();

   delay(200);

 }

**Serial**.println("Nasiel som volnu cestu.");

 dopredu();

}

void zistiVzdialenost() {

 digitalWrite(TRIG\_PIN, LOW);

 delayMicroseconds(2);

 digitalWrite(TRIG\_PIN, HIGH);

 delayMicroseconds(10);

 digitalWrite(TRIG\_PIN, LOW);

 long duration = pulseIn(ECHO\_PIN, HIGH, 25000);

 distance = (duration/2) \* 0.034;

}

void zastav() {

 vypniA();

 vypniB();

 digitalWrite(SPIATOCKA, LOW);

 digitalWrite(PP\_SMEROVKA, LOW);

 digitalWrite(PL\_SMEROVKA, LOW);

 digitalWrite(ZP\_SMEROVKA, LOW);

 digitalWrite(ZL\_SMEROVKA, LOW);

 digitalWrite(P\_BRZDNE, HIGH);

 digitalWrite(L\_BRZDNE, HIGH);

 vbs = millis() + 1500;

 smer = STAT;

}

void smerA(int vpred) {      // vpred = 1 ak chces ist dopredu, vpred = 0 ak chces ist dozadu

 if (vpred != 0) {

   digitalWrite(SMER\_A1, LOW);

   digitalWrite(SMER\_A2, HIGH);

 } else {

   digitalWrite(SMER\_A1, HIGH);

   digitalWrite(SMER\_A2, LOW);

 }

}

void zapniA() {

 digitalWrite(RYCHLOST\_A, HIGH);

}

void vypniA() {

 digitalWrite(RYCHLOST\_A, LOW);

}

void rychlostA(int kolko) {

 analogWrite(RYCHLOST\_A, kolko);

}

void smerB(int vpred) {

 if (vpred != 0) {

   digitalWrite(SMER\_B1, LOW);

   digitalWrite(SMER\_B2, HIGH);

 } else {

   digitalWrite(SMER\_B1, HIGH);

   digitalWrite(SMER\_B2, LOW);

 }

}

void zapniB() {

 digitalWrite(RYCHLOST\_B, HIGH);

}

void vypniB() {

 digitalWrite(RYCHLOST\_B, LOW);

}

void rychlostB(int kolko) {

 analogWrite(RYCHLOST\_B, kolko);

}

## Program na ovládanie robota (na PC)

Trieda HlavneOkno:

**package** robot;  
  
**import** java.awt.Color;  
**import** java.awt.EventQueue;  
**import** java.awt.event.ActionEvent;  
**import** java.awt.event.ActionListener;  
  
**import** javax.swing.ButtonGroup;  
**import** javax.swing.DefaultListModel;  
**import** javax.swing.JButton;  
**import** javax.swing.JCheckBox;  
**import** javax.swing.JFrame;  
**import** javax.swing.JLabel;  
**import** javax.swing.JList;  
**import** javax.swing.JRadioButton;  
**import** javax.swing.JScrollBar;  
**import** javax.swing.JTextField;  
**import** javax.swing.ListModel;  
  
**import** bluetooth.Bluetooth;  
**import** javax.swing.JScrollPane;  
**import** javax.swing.JSlider;  
**import** javax.swing.event.ChangeListener;  
**import** javax.swing.event.ChangeEvent;  
  
**public class** HlavneOkno {  
   
 **private** Bluetooth bluetooth = Bluetooth.priprav();  
 **private** JFrame frmOvladacRobota;  
 **private** JTextField textVzdialenost;  
 **private** DefaultListModel listModelSpravy;  
 **private** JList listSpravy;  
 **private** JScrollPane scrollPaneSpravy;  
 **private** JCheckBox chckbxZastavRolovanie;  
 **private** JCheckBox chckbxZastavPrimanie;  
 **private** JSlider sliderCitlivostLightAssist;  
 **private** JCheckBox chckbxStretavacieSvetla;  
 **private** JCheckBox chckbxDialkoveSvetla;  
 **private** JCheckBox chckbxLightAssist;  
 **private** JCheckBox chckbxDiagnostickeVypisy;  
  
 **public static void** main(String[] args) {  
 EventQueue.invokeLater(**new** Runnable() {  
 **public void** run() {  
 **try** {  
 HlavneOkno window = **new** HlavneOkno();  
 window.frmOvladacRobota.setVisible(**true**);  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 });  
 }  
  
 **public** HlavneOkno() {  
 initialize();  
 Zobrazovac zobrazovac = **new** Zobrazovac(bluetooth, **this**);  
 Thread thread = **new** Thread(zobrazovac);  
 thread.setDaemon(**true**);  
 thread.start();  
 }  
  
 **private void** initialize() {  
 frmOvladacRobota = **new** JFrame();  
 frmOvladacRobota.setTitle(**"Ovl\u00E1da\u010D robota"**);  
 frmOvladacRobota.setForeground(Color.WHITE);  
 frmOvladacRobota.setBounds(100, 100, 1009, 498);  
 frmOvladacRobota.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
   
 JButton btnNewButton = **new** JButton(**"STOP"**);  
 btnNewButton.setToolTipText(**"zastavit robota"**);  
 btnNewButton.setBounds(840, 367, 78, 44);  
 btnNewButton.setBackground(Color.RED);  
 btnNewButton.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 bluetooth.posli(**"E"**);  
 }  
 });  
 frmOvladacRobota.getContentPane().setLayout(**null**);  
 frmOvladacRobota.getContentPane().add(btnNewButton);  
   
 JButton btnVl = **new** JButton(**"vl"**);  
 btnVl.setToolTipText(**"vlavo"**);  
 btnVl.setBounds(776, 367, 64, 44);  
 btnVl.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 bluetooth.posli(**"C"**);  
 }  
 });  
 btnVl.setBackground(Color.YELLOW);  
 frmOvladacRobota.getContentPane().add(btnVl);  
   
 JButton btnVp = **new** JButton(**"vp"**);  
 btnVp.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 bluetooth.posli(**"B"**);  
 }  
 });  
 btnVp.setToolTipText(**"vpravo"**);  
 btnVp.setBounds(919, 367, 64, 44);  
 btnVp.setBackground(Color.YELLOW);  
 frmOvladacRobota.getContentPane().add(btnVp);  
   
 JButton btnDz = **new** JButton(**"dz"**);  
 btnDz.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 bluetooth.posli(**"D"**);  
 }  
 });  
 btnDz.setToolTipText(**"dozadu"**);  
 btnDz.setBounds(850, 408, 64, 44);  
 btnDz.setBackground(Color.YELLOW);  
 frmOvladacRobota.getContentPane().add(btnDz);  
   
 JButton btnDp = **new** JButton(**"dp"**);  
 btnDp.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 bluetooth.posli(**"A"**);  
 }  
 });  
 btnDp.setToolTipText(**"dopredu"**);  
 btnDp.setBounds(845, 325, 64, 44);  
 btnDp.setBackground(Color.YELLOW);  
 frmOvladacRobota.getContentPane().add(btnDp);  
   
 textVzdialenost = **new** JTextField();  
 textVzdialenost.setBounds(167, 429, 599, 23);  
 textVzdialenost.setEditable(**false**);  
 frmOvladacRobota.getContentPane().add(textVzdialenost);  
 textVzdialenost.setColumns(10);  
   
 JLabel lblVzdialenostOdPrekazky = **new** JLabel(**"Vzdialenos\u0165 od prek\u00E1\u017Eky:"**);  
 lblVzdialenostOdPrekazky.setBounds(10, 430, 165, 21);  
 frmOvladacRobota.getContentPane().add(lblVzdialenostOdPrekazky);  
   
 **final** JRadioButton rdbtnR1 = **new** JRadioButton(**"najr\u00FDchlej\u0161ie"**);  
 rdbtnR1.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR1.isSelected()) {  
 bluetooth.posli(**"H"**);  
 }  
 }  
 });  
 rdbtnR1.setToolTipText(**"zapnut rychlost \"najrychlejsie\""**);  
 rdbtnR1.setBounds(10, 257, 107, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR1);  
   
 **final** JRadioButton rdbtnR2 = **new** JRadioButton(**"ve\u013Emi r\u00FDchlo"**);  
 rdbtnR2.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR2.isSelected()) {  
 bluetooth.posli(**"I"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnR2.setToolTipText(**"zapn\u00FA\u0165 r\u00FDchlos\u0165 \"ve\u013Emi r\u00FDchlo\""**);  
 rdbtnR2.setBounds(119, 257, 140, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR2);  
   
 **final** JRadioButton rdbtnR3 = **new** JRadioButton(**"r\u00FDchlo"**);  
 rdbtnR3.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR3.isSelected()) {  
 bluetooth.posli(**"J"**);  
 } **else** {  
 }  
   
 }  
 });  
 rdbtnR3.setToolTipText(**"zapn\u00FA\u0165 r\u00FDchlos\u0165 \"r\u00FDchlo\""**);  
 rdbtnR3.setBounds(261, 257, 87, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR3);  
   
 **final** JRadioButton rdbtnR4 = **new** JRadioButton(**"viac ne\u017E stredne"**);  
 rdbtnR4.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR4.isSelected()) {  
 bluetooth.posli(**"K"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnR4.setToolTipText(**"zapn\u00FA\u0165 r\u00FDchlos\u0165 \"viac ne\u017E stredne r\u00FDchlo\""**);  
 rdbtnR4.setBounds(359, 257, 147, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR4);  
   
 **final** JRadioButton rdbtnR5 = **new** JRadioButton(**"stredne"**);  
 rdbtnR5.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR5.isSelected()) {  
 bluetooth.posli(**"L"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnR5.setToolTipText(**"zapnut rychlost \"stredne\""**);  
 rdbtnR5.setBounds(10, 285, 72, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR5);  
   
 **final** JRadioButton rdbtnR6 = **new** JRadioButton(**"menej ne\u017E stredne"**);  
 rdbtnR6.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR6.isSelected()) {  
 bluetooth.posli(**"M"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnR6.setToolTipText(**"zapn\u00FA\u0165 r\u00FDchlos\u0165 \"menej ne\u017E stredne\""**);  
 rdbtnR6.setBounds(119, 285, 140, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR6);  
   
 **final** JRadioButton rdbtnR7 = **new** JRadioButton(**"pomaly"**);  
 rdbtnR7.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR7.isSelected()) {  
 bluetooth.posli(**"N"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnR7.setToolTipText(**"zapn\u00FA\u0165 r\u00FDchlos\u0165 \"pomaly\""**);  
 rdbtnR7.setBounds(261, 285, 87, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR7);  
   
 **final** JRadioButton rdbtnR8 = **new** JRadioButton(**"najpomal\u0161ie"**);  
 rdbtnR8.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnR8.isSelected()) {  
 bluetooth.posli(**"O"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnR8.setToolTipText(**"zapn\u00FA\u0165 r\u00FDchlos\u0165 \"najpomal\u0161ie\""**);  
 rdbtnR8.setBounds(359, 285, 147, 37);  
 frmOvladacRobota.getContentPane().add(rdbtnR8);  
   
 **final** JRadioButton rdbtnAcp = **new** JRadioButton(**"Vyh\u00FDbanie sa prek\u00E1\u017Ekam doprava"**);  
 rdbtnAcp.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnAcp.isSelected()) {  
 bluetooth.posli(**"P"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnAcp.setToolTipText(**""**);  
 rdbtnAcp.setBounds(10, 337, 416, 23);  
 frmOvladacRobota.getContentPane().add(rdbtnAcp);  
   
 **final** JRadioButton rdbtnAcl = **new** JRadioButton(**"Vyh\u00FDbanie sa prek\u00E1\u017Ekam do\u013Eava"**);  
 rdbtnAcl.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnAcl.isSelected()) {  
 bluetooth.posli(**"Q"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnAcl.setToolTipText(**"zapnut Acc dolava"**);  
 rdbtnAcl.setBounds(10, 362, 423, 23);  
 frmOvladacRobota.getContentPane().add(rdbtnAcl);  
   
 **final** JRadioButton rdbtnAcof = **new** JRadioButton(**"Vypn\u00FA\u0165 vyh\u00FDbanie sa prek\u00E1\u017Ekam"**);  
 rdbtnAcof.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (rdbtnAcof.isSelected()) {  
 bluetooth.posli(**"R"**);  
 } **else** {  
 }  
 }  
 });  
 rdbtnAcof.setToolTipText(**""**);  
 rdbtnAcof.setBounds(10, 388, 404, 23);  
 frmOvladacRobota.getContentPane().add(rdbtnAcof);  
   
 ButtonGroup group3 = **new** ButtonGroup();  
 group3.add(rdbtnAcl);  
 group3.add(rdbtnAcp);  
 group3.add(rdbtnAcof);  
   
 **final** JCheckBox chckbxUltzSenzor = **new** JCheckBox(**"ultrazvukov\u00FD senzor"**);  
 chckbxUltzSenzor.setSelected(**true**);  
 chckbxUltzSenzor.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (chckbxUltzSenzor.isSelected()) {  
 bluetooth.posli(**"F"**);  
 } **else** {  
 bluetooth.posli(**"G"**);  
 }  
 }  
 });  
 chckbxUltzSenzor.setBounds(10, 231, 165, 23);  
 frmOvladacRobota.getContentPane().add(chckbxUltzSenzor);  
   
 scrollPaneSpravy = **new** JScrollPane();  
 scrollPaneSpravy.setBounds(10, 10, 973, 214);  
   
 frmOvladacRobota.getContentPane().add(scrollPaneSpravy);  
  
 listModelSpravy = **new** DefaultListModel();  
 listSpravy = **new** JList(listModelSpravy);  
 scrollPaneSpravy.setViewportView(listSpravy);  
   
 chckbxZastavRolovanie = **new** JCheckBox(**"zastav rolovanie"**);  
 chckbxZastavRolovanie.setBounds(863, 231, 120, 23);  
 frmOvladacRobota.getContentPane().add(chckbxZastavRolovanie);  
   
 chckbxZastavPrimanie = **new** JCheckBox(**"zastav prij\u00EDmanie"**);  
 chckbxZastavPrimanie.setBounds(710, 231, 135, 23);  
 frmOvladacRobota.getContentPane().add(chckbxZastavPrimanie);  
   
 ButtonGroup group2 = **new** ButtonGroup();  
 group2.add(rdbtnR1);  
 group2.add(rdbtnR2);  
 group2.add(rdbtnR5);  
 group2.add(rdbtnR6);  
 group2.add(rdbtnR8);  
 group2.add(rdbtnR4);  
 group2.add(rdbtnR7);  
 group2.add(rdbtnR3);  
   
 chckbxStretavacieSvetla = **new** JCheckBox(**"stret\u00E1vacie svetl\u00E1"**);  
 chckbxStretavacieSvetla.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (chckbxStretavacieSvetla.isSelected()) {  
 bluetooth.posli(**"S"**);  
 chckbxLightAssist.setSelected(**false**);  
 } **else** {  
 bluetooth.posli(**"T"**);  
 }  
 }  
 });  
 chckbxStretavacieSvetla.setBounds(710, 250, 135, 50);  
 frmOvladacRobota.getContentPane().add(chckbxStretavacieSvetla);  
   
 chckbxDialkoveSvetla = **new** JCheckBox(**"dia\u013Ekov\u00E9 svetl\u00E1"**);  
 chckbxDialkoveSvetla.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (chckbxDialkoveSvetla.isSelected()) {  
 bluetooth.posli(**"U"**);  
 chckbxStretavacieSvetla.setSelected(**true**);  
 chckbxLightAssist.setSelected(**false**);  
 } **else** {  
 bluetooth.posli(**"V"**);  
 }  
 }  
 });  
 chckbxDialkoveSvetla.setBounds(863, 250, 120, 50);  
 frmOvladacRobota.getContentPane().add(chckbxDialkoveSvetla);  
   
 chckbxLightAssist = **new** JCheckBox(**"Light assist"**);  
 chckbxLightAssist.addActionListener(**new** ActionListener() {  
 **public void** actionPerformed(ActionEvent e) {  
 **if** (chckbxLightAssist.isSelected()) {  
 posliLightAssist();  
 chckbxStretavacieSvetla.setSelected(**false**);  
 chckbxDialkoveSvetla.setSelected(**false**);  
 } **else** {  
 bluetooth.posli(**"X"**);  
 }  
 }  
 });  
 chckbxLightAssist.setBounds(549, 250, 159, 50);  
 frmOvladacRobota.getContentPane().add(chckbxLightAssist);  
   
 sliderCitlivostLightAssist = **new** JSlider();  
 sliderCitlivostLightAssist.addChangeListener(**new** ChangeListener() {  
 **public void** stateChanged(ChangeEvent e) {  
 **if** (chckbxLightAssist.isSelected()) {  
 posliLightAssist();  
 }  
 }  
 });  
 sliderCitlivostLightAssist.setMinimum(0);  
 sliderCitlivostLightAssist.setMaximum(9);  
 sliderCitlivostLightAssist.setBounds(561, 307, 147, 31);  
 frmOvladacRobota.getContentPane().add(sliderCitlivostLightAssist);  
   
 chckbxDiagnostickeVypisy = **new** JCheckBox(**"diagnostick\u00E9 v\u00FDpisy"**);  
 chckbxDiagnostickeVypisy.setSelected(**true**);  
 chckbxDiagnostickeVypisy.setBounds(549, 231, 140, 23);  
 frmOvladacRobota.getContentPane().add(chckbxDiagnostickeVypisy);  
 }  
   
 **private void** posliLightAssist() {  
 **char** citlivost = (**char**) (**'0'** + sliderCitlivostLightAssist.getValue());   
 bluetooth.posli(**"W"** + Character.valueOf(citlivost));  
 }  
  
 **public void** prisielRiadok(String riadok) {  
 String x = **null**;  
 **int** i = riadok.indexOf(**"[D="**);  
 **if** (i >= 0) {  
 **int** j = riadok.indexOf(**" "**);  
 **if** (j >= 0) {  
 x = riadok.substring(i+3, j);  
 textVzdialenost.setText(x + **" cm"**);  
 }  
 }  
   
 **if** (!chckbxZastavPrimanie.isSelected() &&  
 (chckbxDiagnostickeVypisy.isSelected() || !riadok.startsWith(**"["**))) {  
 listModelSpravy.addElement(riadok);  
 **if** (!chckbxZastavRolovanie.isSelected()) {  
 JScrollBar sb = scrollPaneSpravy.getVerticalScrollBar();  
 sb.setValue(sb.getMaximum());  
 }  
 }  
 }  
}

Trieda Zobrazovac:

**package** robot;  
  
**import** bluetooth.Bluetooth;  
  
**public class** Zobrazovac **implements** Runnable {  
  
 **private** Bluetooth **bluetooth**;  
 **private** HlavneOkno **hlavneOkno**;  
  
 **public** Zobrazovac(Bluetooth bluetooth, HlavneOkno hlavneOkno) {  
 **this**.**bluetooth** = bluetooth;  
 **this**.**hlavneOkno** = hlavneOkno;  
 }  
  
 @Override  
 **public void** run() {  
 **for** (;;) {  
 String riadok = **bluetooth**.prijmi();  
 **if** (riadok == **null**) {  
 **return**;  
 }  
 **hlavneOkno**.prisielRiadok(riadok);  
 }  
 }  
}

Trieda Bluetooth:

**package** bluetooth;  
  
**import** java.io.BufferedReader;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.io.InputStreamReader;  
**import** java.io.OutputStream;  
**import** java.io.OutputStreamWriter;  
**import** java.io.PrintWriter;  
  
**import** javax.microedition.io.Connector;  
**import** javax.microedition.io.StreamConnection;  
  
**public class** Bluetooth {  
  
 **private** PrintWriter **writer**;  
 **private** BufferedReader **reader**;  
 **private** StreamConnection **streamConnection**;  
  
 **private boolean vypisy** = **true**;  
  
 **private** Bluetooth(String url) **throws** IOException {  
 System.***out***.println(**"Pripajam sa na "** + url);  
 **streamConnection** = (StreamConnection) Connector.open(url);  
 OutputStream outStream = **streamConnection**.openOutputStream();  
 InputStream inStream = **streamConnection**.openInputStream();  
 **writer** = **new** PrintWriter(**new** OutputStreamWriter(outStream));  
 **reader** = **new** BufferedReader(**new** InputStreamReader(inStream));  
 System.***out***.println(**"Uspesne pripojeny."**);  
 }  
  
 **public static** Bluetooth priprav() {  
 String url = **"btspp://201411272918:1;authenticate=false;encrypt=false;master=false"**;  
 **try** {  
 **return new** Bluetooth(url);  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **return null**;  
 }  
 }  
  
 **public void** posli(String s) {  
 **if** (**vypisy**) {  
 System.***out***.println(**"Posielam: '"** + s + **"'"**);  
 }  
 **writer**.print(s);  
 **writer**.flush();  
 }  
  
 **public** String prijmi() {  
 **try** {  
 String s = **reader**.readLine();  
 **if** (**vypisy**) {  
 System.***out***.println(**"Prijal som: '"** + s + **"'"**);  
 }  
 **return** s;  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **return null**;  
 }  
 }  
  
 **public void** zatvor() {  
 **try** {  
 **streamConnection**.close();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
}

1. <https://github.com/janmederly/rp6/blob/master/arduino/p1_diody_ovladane_bluetoothom> [↑](#footnote-ref-1)
2. <https://github.com/janmederly/rp6/tree/master/arduino/p2_diody_ovladane_bluetoothom_3_farebna_dioda> [↑](#footnote-ref-2)
3. <https://github.com/janmederly/rp6/tree/master/arduino/p3_bluetooth_robot> [↑](#footnote-ref-3)
4. <https://github.com/janmederly/rp6/tree/master/arduino/p4_robot_s_asistentom_narazu> [↑](#footnote-ref-4)