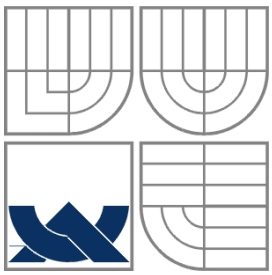


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

NÁVRH METODY DETEKCE VZÁJEMNÉ POLOHY VOZIDEL V AUTONOMNÍM KONVOJI

DESIGN OF METHOD FOR POSITION DETECTION OF AUTONOMOUS CONVOY VEHICLES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **JAKUB KRYSL**
AUTHOR

VEDOUCÍ PRÁCE **ING. STANISLAV VĚCHET, PH.D.**
SUPERVISOR

BRNO 2013

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2012/2013

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Jakub Krysl

který/která studuje v **bakalářském studijním programu**

obor: **Aplikovaná informatika a řízení (3902R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Návrh metody detekce vzájemné polohy vozidel v autonomním konvoji

v anglickém jazyce:

Desing of method for position detection of autonomous convoy vehicles

Stručná charakteristika problematiky úkolu:

Jízda autonomních vozidel v konvoji s prvním vozidlem řízeným operátorem patří k předním problémům mobilní robotiky.
Hlavním cílem práce je průzkum metod vhodných k detekci vzájemné polohy vozidel v konvoji. Na základě tohoto průzkumu bude vybrána metoda k podrobnější analýze a testům na reálném modelu
v měřítku 1:10.

Cíle bakalářské práce:

1) Seznamte se s aktuálně používanými metodami pro jízdu autonomních vozidel v konvoji.

2) Podrobně prostudujte vlastnosti jednotlivých metod.

3) Na základě rešeršní studie vyberte nejvhodnější metodu.

4) Vybranou metodu experimentálně otestujte.

Seznam odborné literatury:

[1] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series). Intelligent robotics and autonomous agents. The MIT Press, August 2005.
[2] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. Principles of Robot Motion: Theory, Algorithms, and Implementations (Intelligent Robotics and Autonomous Agents). The MIT Press, June 2005.

Vedoucí bakalářské práce: Ing. Stanislav Věchet, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2012/2013.

V Brně, dne 15.11.2012

L.S.

Ing. Jan Roupec, Ph.D. Ředitel ústavu	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c. Děkan fakulty

ABSTRAKT

Tato práce se zabývá metodami řízení autonomních vozidel v konvoji a testováním vybrané metody tohoto řízení na reálném modelu v měřítku 1:10. Jako platforma pro testování metod detekce vedoucího vozidla slouží BeagleBoard xM.

ABSTRACT

This thesis deals with autonomous convoy vehicles control methods. A real 1:10 scale model was choosen as a control and testing platform. As the main computation unit serves the BeagleBoard xM which runs the used detection method.

KLÍČOVÁ SLOVA

Řízení, konvoj, BeagleBoard, Ubuntu, robotika, detekce, OpenCV.

KEYWORDS

Controlling, convoy, BeagleBoard, Ubuntu, robotics, detection, OpenCV.

PROHLÁŠENÍ O ORIGINALITĚ

Já Jakub Krysl, prohlašuji, že jsem bakalářskou práci vypracoval samostatně a že jsem uvedl všechny použité prameny a literaturu.

V Brně dne 24.5.2013 ……

BIBLIOGRAFICKÁ CITACE

KRYSL, J. *Návrh metody detekce vzájemné polohy vozidel v autonomním konvoji.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2013. 43 s. Vedoucí bakalářské práce Ing. Stanislav Věchet, Ph.D..

PODĚKOVÁNÍ

Rád bych touto cestou poděkoval vedoucímu své práce Ing. Stanislavu Věchetovi, Ph.D. za cenné rady a za pomoc při tvorbě této práce.

OBSAH:

1 ÚVOD

Tato bakalářská práce se zabývá metodami řízení autonomních vozidel v konvoji. Jejím cílem je seznámit se s aktuálně používanými metodami tohoto řízení a následně jednu metodu vybrat a experimentálně otestovat na modelu sestaveném ze 2 RC modelů v měřítku 1:10 jedoucích v konvoji, kdy hlavní řízení je realizováno na jednodeskovém počítači BeagleBoard xM. Vzhledem k tomu, že tématika autonomních konvojů je velmi obsáhlá, je hlavní část práce věnována řešení problému sledování vedoucího vozidla.

2 AUTONOMNÍ ŘÍZENÍ VOZIDLA

V současné době je autonomní řízení vozidla v konvoji odvětvím ve vývoji, kterým se zabývají všichni velcí výrobci automobilů. Cílem je nejen zvýšit pohodlí řidiče, ale současně zvýšit bezpečnost silniční dopravy a snížit její energetickou náročnost a tudíž i náročnost na životní prostředí.

K tomuto problému se v současné době přistupuje ze 2 směrů. Prvním je autonomní řízení vozidla v konvoji s prvním vozidlem řízeným řidičem. Druhým je autonomní řízení pouze jednoho vozidla, tudíž kompletní nahrazení role řidiče. Abych mohl zvážit výhody a nevýhody jednotlivých přístupů, musím se zabývat oběma směry. V další kapitole popíši vybrané řešení řízení a důvody výběru toto řešení.

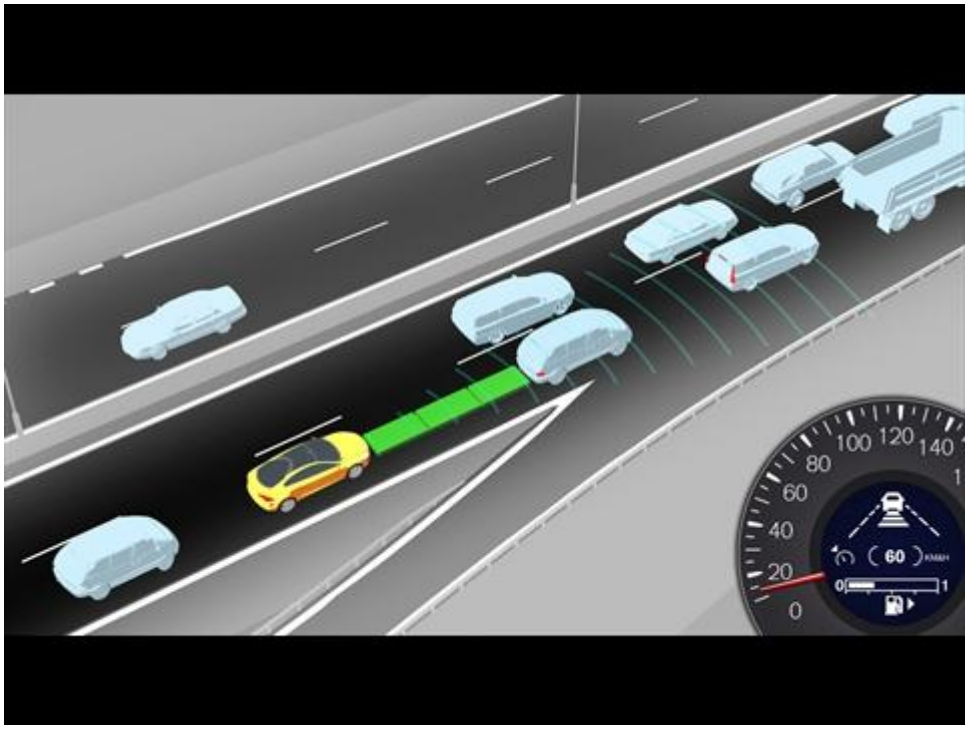
2.1 Řízení vozidla v konvoji

Autonomní řízení vozidla v konvoji je v základě pouze následování vozidla předchozího. Při tomto druhu řízení vozidlo nemusí sledovat okolí, neboť veškeré sledování okolí nechává na vozidle před sebou respektive na prvním vozidle v konvoji. Díky tomu je tento druh řízení jednodušší oproti plně autonomnímu řízení jednoho vozidla a je ho tudíž snazší i široce implementovat.

Průkopnou technologií v této oblasti je adaptivní tempomat, díky kterému se tento způsob řízení začíná vyskytovat u dražších a luxusnějších osobních automobilů. Tento systém vůbec nekomunikuje s vozem před sebou, a proto se vyvíjí kooperativní adaptivní tempomat, který tento nedostatek odstraní a umožní nástup pokročilejších technologií v tomto způsobu řízení. Na systém kooperativního adaptivního tempomatu navazuje projekt SARTRE, podporu pro celkovou infrastrukturu přenosu informací mezi vozy vytváří projekt CVIS a ověření schopnosti vozu urazit autonomně velké vzdálenosti realizoval projekt VIAC.

2.1.1 Adaptivní tempomat

Adaptivní tempomat (ACC – Adaptive Cruise Control) je systém řízení automatického vozidla, který nejenže udržuje rychlost jako klasický tempomat, ale tuto rychlost přizpůsobuje autu před sebou. Tento systém je na trhu již několik let a ve výbavě ho nabízí většina výrobců aut. Možnosti ACC se liší jednak podle výrobce – například Toyota nabízí nastavitelnost vzdálenosti od vozidla před sebou – nebo podle možností vozidla. Automobily s automatickou převodovkou jsou schopny automaticky se zastavit a i rozjet, kdežto ACC u automobilů s manuální převodovkou funguje od rychlosti 30km/h. [3][4]



Obr. 1 Adaptivní tempomat [6]

Celý systém funguje jako normální tempomat až do okamžiku, kdy senzory zachytí před vozem překážku. V té chvíli je rychlost snížena na rychlost objektu před vozidlem. Tedy pokud je to automobil, je rychlost vyrovnána. Ve chvíli, kdy překážka zmizí, zrychlí auto zpět na nastavenou rychlost. Použité senzory jsou laserové nebo radarové. Senzory na bázi laseru mají nevýhodu, že nejsou použitelné v prostředí se sníženou viditelností (prach, sníh, prudký déšť) a také mají problém se zaprášenými auty či jakýmikoliv povrchy se sníženou viditelností. [5]

2.1.2 Kooperativní adaptivní tempomat

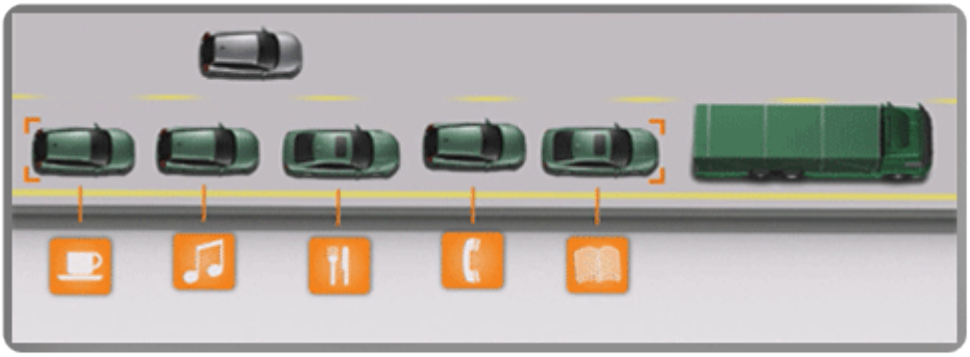
Kooperativní adaptivní tempomat (CACC – Cooperative Adaptive Cruise Control) je další krok ve vývoji autonomního řízení automobilu hned po adaptivním tempomatu. Celý systém je vylepšen o komunikaci mezi jednotlivými vozidly. Oproti ACC toto řízení poskytuje výhodu plynulejší jízdy, kdy systém dokáže předvídat překážky na cestě díky informacím získaným vozy před ním. Také to vede k celkově plynulejšímu a bezpečnějšímu provozu a možnosti menší vzdálenosti mezi vozidly a tudíž úspoře energie díky snížení odporu vzduchu. A také díky menším mezerám mezi vozidly a malými rozdíly rychlostí mezi jednotlivými vozidly dokáže CACC zvýšit kapacitu dálnic, jak potvrdil test na MIXIC modelu. [7]

2.1.3 SARTRE

Tento projekt byl financován Evropskou komisí na základě programu Framework 7. Jeho cílem bylo zpříjemnit dálkové cesty pro řidiče osobních aut a současně omezit škodlivý dopad osobní dopravy na životní prostředí. Název projektu je zkratkou pro Safe Road Trains for The Enviroment, volně přeloženo bezpečné silniční vlaky pro životní prostředí. V čele projektu stála společnost Ricardo UK Ltd, která řídila spolupráci společností a institutů Idiada and Robotiker-Tecnalia of Spain, Institut for Kraftfahrwesen Aachen (IKA) of Germany, SP Technical Research Institute of Sweden, Volvo Car Corporation a Volvo Technology of Sweden. [8]

Projekt trval od září 2009 do září 2012 a měl 6 etap. SARTE byl zaměřen na vytvoření silničního vlaku, kdy vedoucí vůz je nákladní auto řízené profesionálním řidičem a osazené systémem SARTRE, který umožní několika osobním či nákladním vozidlům tento řídící vůz následovat zcela

automaticky. V základě se jedná o další krok ve vývoji automatického řízení vozidla hned po kooperativním adaptivním tempomatu, na který tento projekt navazuje. [8]



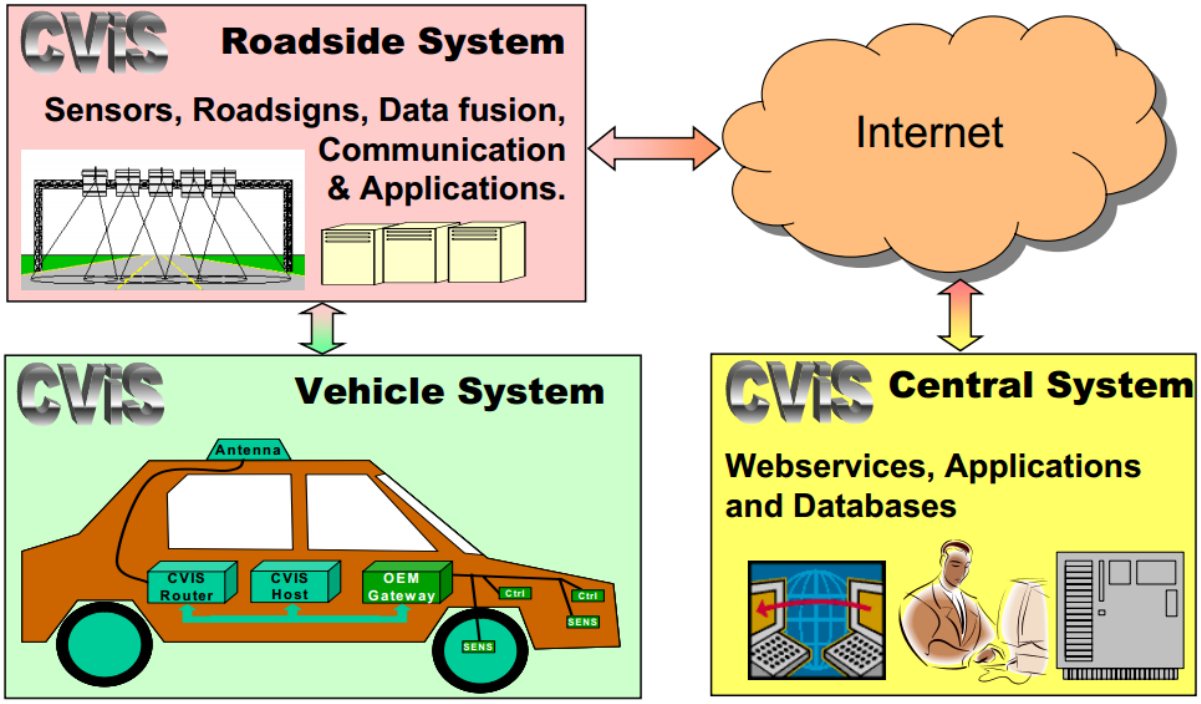
Obr. 2 Schéma projektu SARTRE [8]

Dle závěrečné zprávy [9] byl projekt úspěšně dokončen v roce 2012 s výsledkem, že tento druh dopravy lze použít v běžné dopravě na běžné dálnici. Vzhledem k tomu, že jediný výrobce automobilů zapojený do tohoto projektu byl Volvo Car Corporation, byly během testů použity pouze automobily této značky. Testovací konvoj se skládal z řídícího nákladního auta, za ním následovalo další nákladní auto a poté tři osobní automobily značky Volvo (S60, V60 a XC60). Tento celý konvoj se pohyboval autonomně po dálnici rychlostí 90km/h a v některých případech nebyla mezera mezi jednotlivými vozy větší než 4 metry. Jednotlivá autonomně řízená auta se navigovala pomocí opakování pohybů aut před sebou, kdy tento pohyb snímala pomocí vylepšení kamerových, radarových a laserových technologií již implementovaných systémů bezpečnosti a podpory řízení. Tyto systémy jsou například Lane Keeping, City Safety, Adaptive Cruise Control, Park Assist Pilot a Blind Spot Information System. Každé vozidlo je navíc osazeno HMI panelem s touch screenem, kde jsou jednak zobrazeny důležité údaje o konvoji a také lze díky němu vydat jednoduchý povel k zařazení se či opuštění konvoje. Tento panel by též měl být využíván ke komunikaci s ostatními vozy konvoje. [9]

Výhody tohoto systémů řízení jsou především pohodlí řidičů, úspora energie 10-20% díky nižšímu odporu vzduchu způsobenému menší vzdáleností mezi vozidly, plynulejší doprava zapříčiněná menšími rozdíly rychlostí mezi jednotlivými vozidly a vyšší bezpečnost provozu díky kratší reakční době automatických systémů a zkušenostem profesionálního řidiče vedoucí konvoj. Avšak než tento projekt bude možné realizovat v běžném provozu pro běžné řidiče, je potřeba vyřešit několik problémů týkající se bezpečnosti, například vyhýbání se překážce nebo náhlé brzdění. [9]

2.1.4 CVIS

Projekt CVIS – neboli Cooperative Vehicle Infrastracture System – je zaměřen na vyvíjení systému komunikace jednak mezi vozidlem a systémem řízení dopravy a také mezi samotnými vozidly. Cílem tohoto projektu je vytvoření zázemí pro druhotné aplikace směřující ke zlepšení bezpečnosti a komfortu silniční dopravy. [10]



Obr. 3 Základní subsystémy CVIS [11]

Díky tomuto systému může řidič získat nejrychlejší trasu do požadovaného cíle na základě údajů dostupných od ostatních účastníků provozu a systémů řízení provozu. Jakékoliv informace zobrazované na dopravních značkách mohou být zobrazeny na displeji v automobilu. Toto lze využít například při nouzové situaci, kdy řidič je okamžitě informován o nehodě na trase či přijíždějícím vozidlu záchranných složek, což zvýší bezpečnost na obou stranách. Cílem tohoto systému je také kompletní monitoring všech účastníků provozu s přesností až na 1 metr, tedy dokonce pozice vozidla v jízdním pruhu. To má za následek včasné odhalení vozidla jedoucí v protisměru či zjednodušení řízení dopravy a předcházení zácpám. [10]

Ovšem aby tohoto bylo dosaženo, musí být každé vozidlo vybaveno technologií CVIS, tedy nejen nová, ale i stará vozidla. Toho chce tento projekt dosáhnout vytvořením mobilního routeru komunikujícího s celým systémem bezdrátově pomocí mobilních sítí, lokálních Wi-Fi, infračerveného přenosu nebo DSRC. [10]

Tento projekt trval od roku 2006 do roku 2009 a podílelo se na něm 63 firem z 12 zemí. Testování probíhalo v různých lokalitách nalézajících se v 6 zemích Evropské Unie – UK, Švédsku, Francii, Německu, Itálii a Belgii – a od roku 2009 i v Norsku ve městě Trondheim. [10]

2.1.5 VIAC

Laboratoř VisLab se v roce 2010 podařil velký pokrok ve směru řízení vozidla autonomně i v konvoji. Realizovali projekt VIAC (the VisLab Intercontinental Autonomous Challenge), během kterého 4 autonomně řízená vozidla urazila vzdálenost 15 926 km za 100 dní. Cílem tohoto projektu bylo jednak otestovat schopnost autonomně řízeného vozidla urazit tak velkou vzdálenost, ale hlavně jeho schopnosti se pohybovat v běžné dopravě a za jakýchkoliv podmínek, protože během cesty se pohybovali v rozmezí 44°C a 2 900 výškových metrů napříč 9 zeměmi. [12]

Konceptem řízení bylo rozdělení těchto 4 vozidel na páry, ve kterých byl vždy řídící vůz a jeho následovník. Řídící vůz měl za úkol najít idelání trasu pomocí signálu GPS porovnaným s okolím. Pokud by celá trasa byla přesně zmapována a pokryta body GPS, mohl by tento vůz jet plně autonomně. Vzhledem k délce trasy a území, kterým vozy projížděly, nemohla být tato podmínka splněna. Proto první vůz jel autonomně většinu cesty a pokud se dostal do situace, kde se nemohl rozhodnout sám, převzal řízení člověk pouze po dobu zvládání této situace. Druhý vůz byl plně autonomní. Trasu volil 2 způsoby. Pokud byl v dohledu řídící vůz, následoval ho. Pokud neviděl řídící

vůz, navigoval se podle GPS bodů zanechaných řídícím vozem. Každý pár byl po cestě doprovázen ještě nákladním autem poskytujícím zázemí. [12]

Obr. 4 Celý konvoj VIAC [13]

Řízení bylo realizováno pomocí několika typů senzorů. V každém rohu střešní konstrukce se nacházela kamera s barevným obrazem. Všechny 4 kamery měly za úkol zjišťovat překážky, ty vepředu navíc ještě určovaly příkrost srázu a staraly se o detekci vodorovného značení. Mezi předními kamerami byl nainstalován laserový scanner pro navigaci mimo silnici. Byl schopen detekovat jakékoliv překážky či nerovnosti na cestě přímo před vozem. V zadní části střešní konstrukce se nacházela lokalizační jednotka vybavená přijímačem signálu GPS a inerciálními senzory pro úpravu dříve přijatých GPS dat v nepřítomnosti signálu GPS. Za zpětným zrcátkem byl připevněn systém 3 synchronizovaných kamer s celkovým záběrem 180° pro panoramatický pohled. Tyto kamery měly za úkol detekovat a následovat řídící vozidlo i v prudkých zatáčkách. V předním nárazníku byli umístěny 3 laserové scannery. Boční skenovaly 1 vodorovnou rovinu se záběrem 240° každý a dosahem kolem 30 metrů s cílem odhalit veškeré překážky, chodce či jiná vozidla. Střední skener skenoval 4 vodorovné roviny se záběrem 100° a byl schopen najít veškeré překážky či jiná vozidla do vzdálenosti až 100 metrů před vlastním vozidlem. [12]

Celý systém automatického řidiče byl napájen externím solárním panelem a záložními bateriemi. Díky tomu byl výkon vozidla nezměněn. Přesto byla jejich maximální rychlost kolem 60 km/h s dojezdem 100km, protože byla použita vozidla s elektrickým pohonem. Napájena byla z elektrické sítě a pokud ta nebyla dostupná, z externích generátorů. [12]

2.2 Autonomní řízení samostatného vozidla

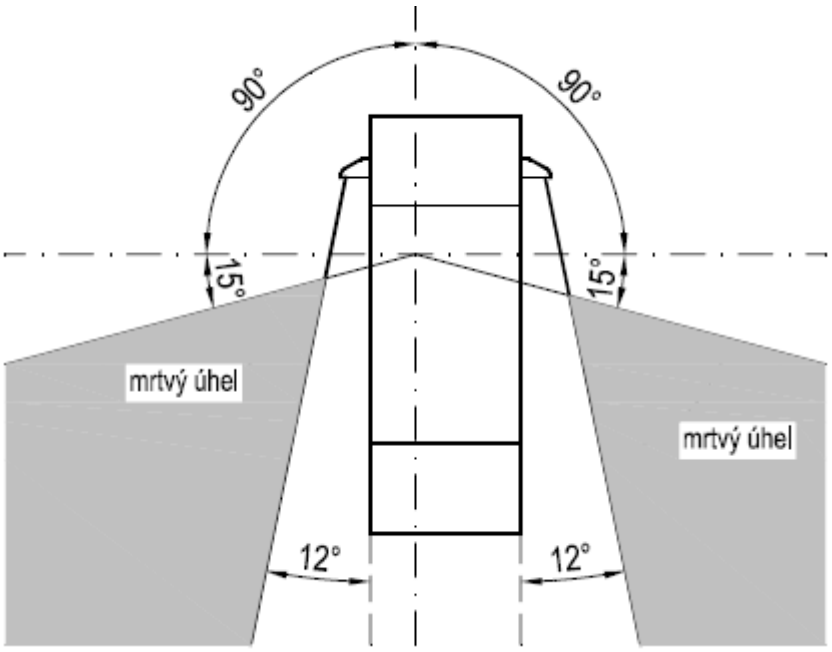
Tento způsob řízení je mnohem složitější než způsob předchozí. Pro výzkumníky zabývající se tímto oborem představuje velkou výzvu a přináší s sebou problémy s touto problematikou spojené. Tyto problémy nejsou pouze technického rázu, tedy samotné řízení vozidla, ale i legislativního. Pokud toto autonomně řízené vozidlo způsobí dopravní havárii nebo se této havárie pouze zúčastní, vyvstává otázka odpovědnosti za danou havárii. Kvůli těmto problémům je vývoj v této oblasti velmi pomalý a velmi nákladný.

Výsledky tohoto vývoje jsou postupně implementovány do současných automobilů. V současné době to jsou systémy, které mají pomoci řidiči v řízení automobilu s cílem primárně zvýšit

bezpečnost jízdy a druhotně i samotný komfort. Jsou to například systémy monitorovaní mrtvého úhlu, sledování jízdního pruhu, parkovací asistent či systém pro vyhnutí se kolizi. Zatím neimplementované kompletní autonomní řízení je stále ve vývoji a potýká se s mnoha problémy, které je třeba vyřešit, než lze tento způsob řízení široce uplatnit v automobilové dopravě. K vývoji v tomto směru přispěly zejména tyto projekty: DARPA Grand Challenge, DARPA Urban Challenge, VIAC a Autonomní vozidlo Google.

2.2.1 Monitorování mrtvého úhlu

Mrtvým úhlem se v automobilovém průmyslu označuje prostor na stranách automobilu, kam řidič nevidí. Řidič vidí na strany automobilu buď přímým pohledem do bočního okénka, nebo pohledem do zpětných zrcátek. Tento problém se řeší primárně asférickými zrcátky nebo sekundárně systémem monitorování mrtvého úhlu. [14]

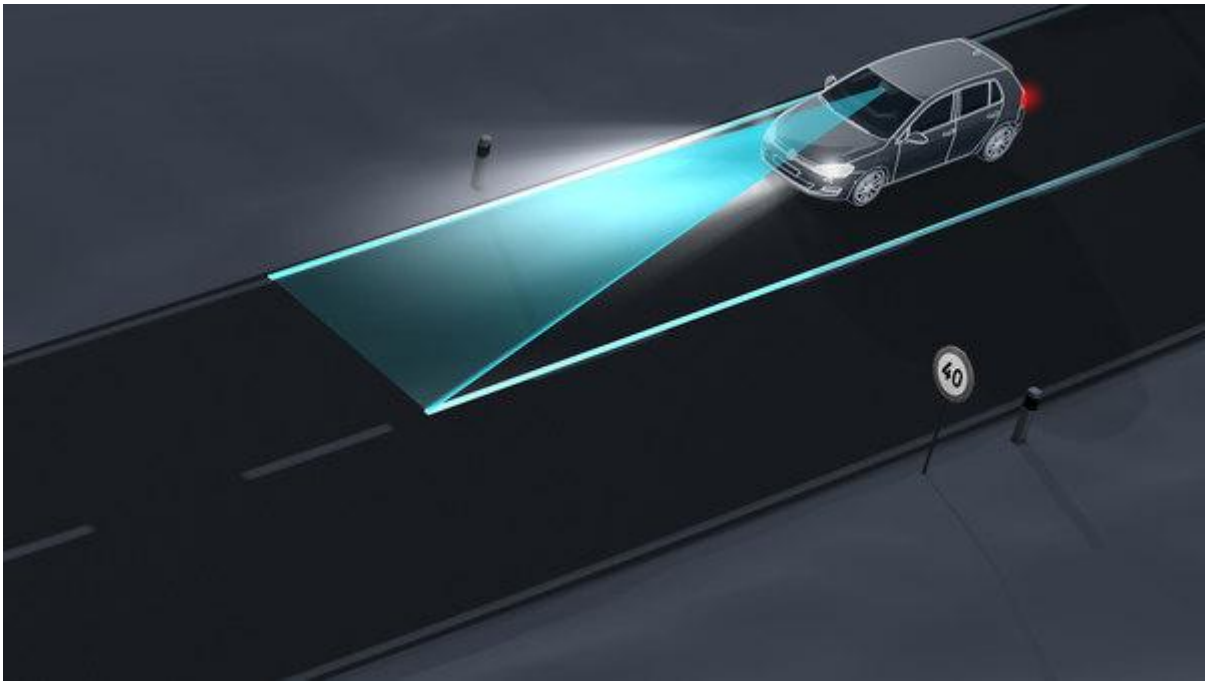


Obr. 5 Mrtvý úhel [15]

Systém monitorování mrtvého úhlu funguje na principu radarových senzorů snímajících objekty – hlavně jiné automobily – v prostoru mrtvého úhlu. Pokud je zde daný objekt zaznamenán, nastává buď aktivní nebo pasivní odezva. Systém pasivní odezvy je například BLIS (Blind Spot Information System), který vyvinula společnost Volvo. Jeho jediným účelem je informování řidiče o objektu v mrtvém úhlu bez jakéhokoliv automatického zásahu do řízení. Přítomnost objekt v daném mrtvém úhlu je indikována rozsvícením světelné diody či jiného značení na daném bočním zrcátku. Systém aktivní odezvy představila společnost Infinity (divize společnosti Nissan zabývající se výrobou luxusních automobilů) pod názvem Blind Spot Intervention System. Pokud auto s tímto systémem se začne pohybovat směrem do prostoru, kde se v mrtvém úhlu nachází jiný automobil, tento systém začne automaticky jemně brzdit kola na opačné straně automobilu a tím ho vrátí do středu svého pruhu a zamezí srážce. [16][17]

2.2.2 Sledování jízdního pruhu

Opouštění jízdního pruhu z důvodu nepozornosti řidiče způsobené především únavou má za následek mnoho dopravních nehod. Proto byl vytvořen systém pro sledování jízdního pruhu a vydání nejdříve pasivního varování a s postupem vývoje i aktivního vrácení vozidla do svého jízdního pruhu. Celý systém funguje na principu kamery umístěné nejčastěji za zpětným zrcátkem. Pomocí výstupu z této kamery je řídící jednotka schopna identifikovat jízdní pruhy za předpokladu, že jsou viditelné.[18]



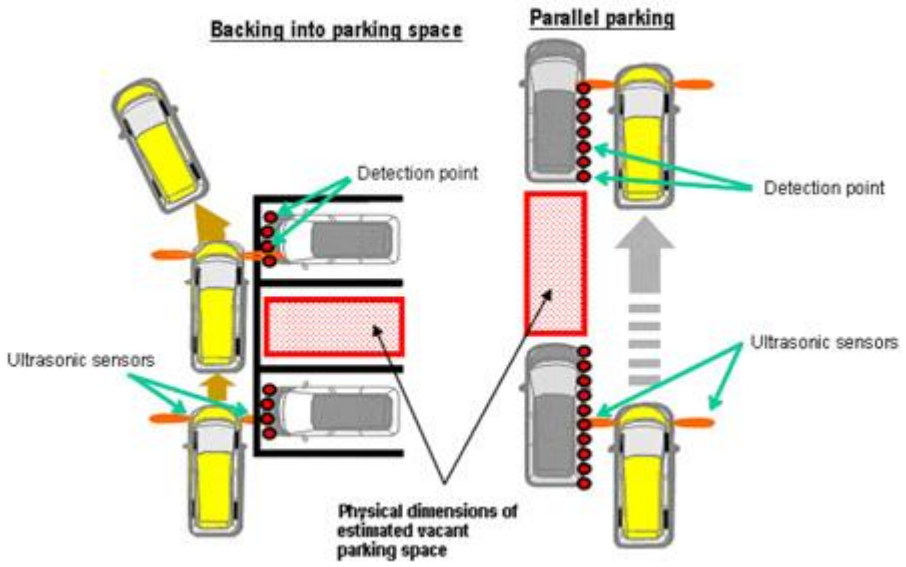
Obr. 6 Lane assist [19]

Pokud řídící jednotka vyhodnotí nechtěné vyjetí z jízdního pruhu, tedy bez použití znamení o změně směru jízdy, zareaguje buď znamením řidiči – vibrováním volantu – nebo aktivním vrácením do jízdního pruhu zatočením volantu směrem zpět do jízdního pruhu. [18]

2.2.3 Parkovací asistent

Pod pojmem parkovací asistent rozumíme jakoukoliv pomoc automatických systémů při parkování. Nejjednodušší parkovací asistenty jsou pouze čidla indukující vzdálenost vozidla od okolních překážek při parkování nebo couvání. V dnešní době je tento typ snadno dostupný a tudíž i dosti rozšířený. Vylepšením této technologie je přidání kamery pro snímání objektů za vozem, kdy výstup z této kamery se zobrazuje na displeji ve středním panelu vozidla, tudíž řidič přesně ví, co se nachází přímo za jeho vozem.

Dalším krokem ve vývoji parkovacího asistentu je automatické vyhledání místa a automatické točení volantem, aby auto správně zajelo do tohoto požadovaného místa. Řidič v tomto případě pouze obsluhuje pedály a v případě manuální převodovky i řadí. Tento systém byl jako první vyvinut firmou Toyota pod názvem Inteligent Parking Assist System (IPAS). Byl aplikován do modelů Toyota Prius v roce 2003 v Japonsku a následně Lexus LS460 v roce 2006 v USA. Systém použitý v roce 2003 měl mnoho chyb – nedokázal si poradit s malými objekty (kočkou, dítětem) či malým parkovacím místem. Tento systém též zvládl zaparkovat pouze podélně. Systém vydaný v roce 2006 měl již tyto chyby opravené a zvládl zaparkovat i pozadu do kolmého místa. Tím se stal široce použitelný a v dnešní době se vyskytuje u luxusnějších aut. [20][21]



Obr. 7 Detekování parkovacího místa [21]

Celý systém funguje díky mnoha ultrazvukovým senzorům umístěným v přední a zadní části vozu, aby mohly snímat objekty kolem celého vozu. Při příjezdu k požadovanému místu parkování zapne řidič parkovací asistent a kolem místa pomalu projede rychlostí pouze několik kilometrů v hodině. Systém na základě dat ze senzorů vyhodnotí vhodnost parkovacího místa a pokud je toto místo vhodné, přebere kontrolu nad řízením a instruuje řidiče, jak řadit a sešlapávat pedály. Řidič musí dle manuálu kontrolovat správnost parkování, protože v případě nehody je právně chyba na straně řidiče. [21]

2.2.4 Systém pro vyhnutí se kolizi

Tento systém využívá několik druhů senzorů pro určení rozdílů rychlosti a vzdálenosti mezi vlastním vozidlem a překážkou před či za vozidlem. Pokud vyhodnotí nebezpečí kolize, varuje o něm řidiče a typ akce nechá na řidiči. Pokud ale k žádné akci nedojde nebo je nedostatečná (například pomalé brzdění), začne automaticky brzdit. Řešení tohoto systému se liší napříč výrobci automobilů, některé dokonce počítají s kolizí jiného auta do vozidla zezadu a jsou schopny v takovém případě zrychlit, pokud tím nevyvstane nebezpečí čelní srážky. [22]

Obr. 8 Zóny systému pro vyhnutí se kolizi [23]

Okolí vozidla je snímáno především kamerou za zpětným zrcátkem a druhotně i radarovými a LIDARovými čidly. Díky těmto senzorům je systém schopen odhalit veškeré možné kolize a včasným upozorněním řidiče či přebráním iniciativy tyto kolize odvrátit či snížit jejich následky. Dle studie [24] vydané Evropskou Komisí lze předejít až 5000 úmrtím a 50 000 vážným zraněním jen v Evropské Unii, pokud by tímto systémem byla vybavena všechna auta. Na základě této studie bylo rozhodnuto, že tímto systémem budou od roku 2013 vybaveny všechny nové typy aut a od roku 2015 všechna nová auta na území Evropské Unie. [22]

2.2.5 DARPA Grand Challenge

Agentura amerického ministerstva obrany DARPA (Defense Advanced Research Projects Agency) je primárně zaměřena na získání a udržení technologického náskoku amerických vojenských sil. V praxi se zaměřuje na podporování malých výzkumných projektů především univerzitních týmů. Projekt této agentury s názvem DARPA Grand Challenge byl veden formou soutěže v letech 2004 a 2005 v Mohavské poušti. Cílem této soutěže bylo nastartovat vývoj na poli plně autonomních vozidel schopných projet danou trasu v různorodém terénu v limitovaném čase. První vůz v cíli měl vyhrát cenu 1 milion $ v prvním ročníku a 2 miliony $ v druhém. [25]

Obr. 9 Logo DARPA Grand Challenge [28]

Prvního ročníku se zúčastnilo 15 týmů, ale žádné vozidlo nedojelo do cíle. Celková trasa závodu činila 240 km. Nejdále se dostalo vozidlo „Sandstorm“, které ujelo 11,9 km. Toto vozidlo bylo osazeno 3 pevnými LIDARy, jedním otočným LIDARem na střeše, raderem, 2 kamerami a samozřejmě GPS. I přes neúspěch vozidel shledala DARPA tento ročník úspěšný vzhledem k vlně popularity a zájmu, kterou tato soutěž přinesla. A proto byl vypsán druhý ročník s předpokladem, že minimálně jedno vozidlo do cíle dojede. [26]

V druhém ročníku soutěžilo celkem 23 týmů na trase dlouhé 212 km. Do cíle tohoto ročníku úspěšně dojelo 5 vozidel, 4 z nich v časovém limitu 10 hodin. Nejrychlejší bylo vozidlo „Stanley“ ze Stanfordské Univerzity. Základem konstrukce bylo auto značky Volkswagen Touareg s 5 senzory LIDAR na střeše s dosahem 80 metrů pro sestavení kompletního 3D modelu okolí kolem vozu, do kterého byly promítnuty body z GPS navigace. Následně byla vybrána nejvhodnější trasa jízdy. Tento systém byl doplněn ještě vstupem z kamery pro zjištění prostoru trasy vzdálenější než 80 metrů. Všechna data byla zpracována 5 počítači umístěnými v kufru auta osazenými 1,8 GHz Pentium M procesory. Všechny počítače běžely na různých verzích operačního systému Linux. Tento vůz dokončil celý závod s časem 6 hodin, 54 minut. [27]

2.2.6 DARPA Urban Challenge

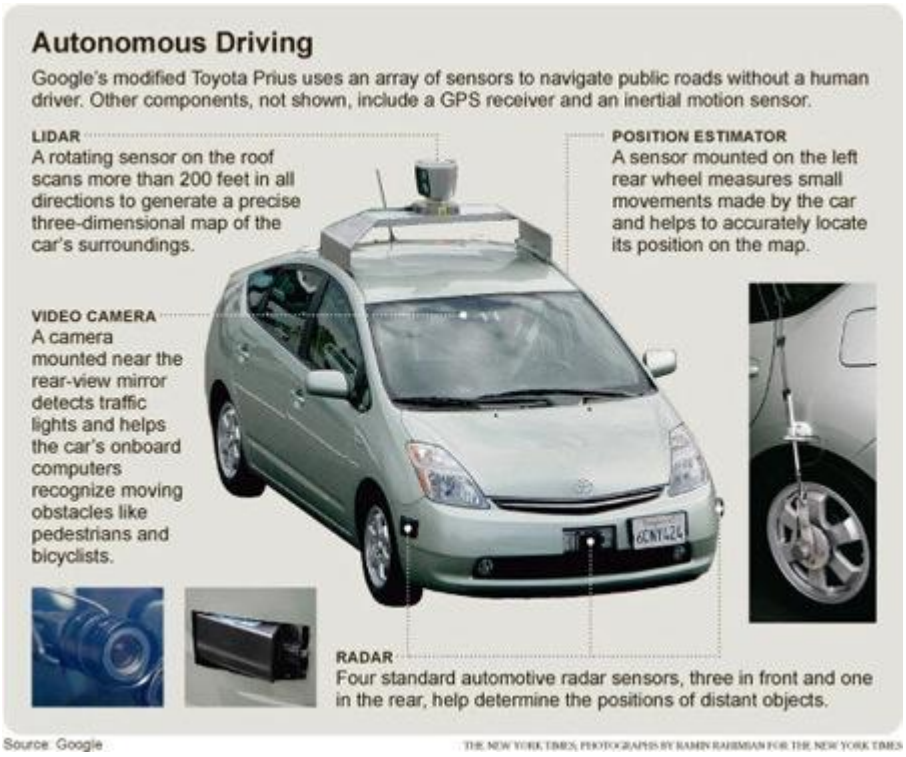
Po úspěchu DARPA Grand Challenge vypsala DARPA novou soutěž s názvem DARPA Urban Challenge. Trasa byla dlouhá 89 kilometrů v zastavěné oblasti v Kalifornii na základně amerických vzdušných sil. Limit pro projetí této trasy byl 6 hodin s tím, že vozy musí dodržovat pravidla silničního provozu. Celá trasa byla koncipována tak, aby jednotlivá vozidla po cestě neustále potkávala jiné vozy a musela řešit problematiku s tímto spojenou. [29]

Z původně přihlášených 89 týmů bylo k závodu přizváno pouze 35. Před samotným závodem se po dobu 8 dnů testovala schopnost vozů se závodu zúčastnit a dodržovat daná pravidla. Původně mělo být vybráno 20 týmů s nejlepšími výsledky, ale pouze 11 týmů nakonec postoupilo k samotnému závodu především kvůli problémům s dodržováním bezpečnosti při jízdě. Závod nakonec dokončilo 6 týmů, z nichž 4 v požadovaném časovém limitu. Výherní automobil Chevy Tahoe „Boss“ týmu Tartan Racing dokončil celý závod s časem 4 hodiny a 10 minut. Tento vůz byl osazen více než tuctem laserových senzorů, kamer a radarů pro získání přesného obrazu světa. [29][30]

Obr. 10 Výherní vůz "Boss" během závodu [31]

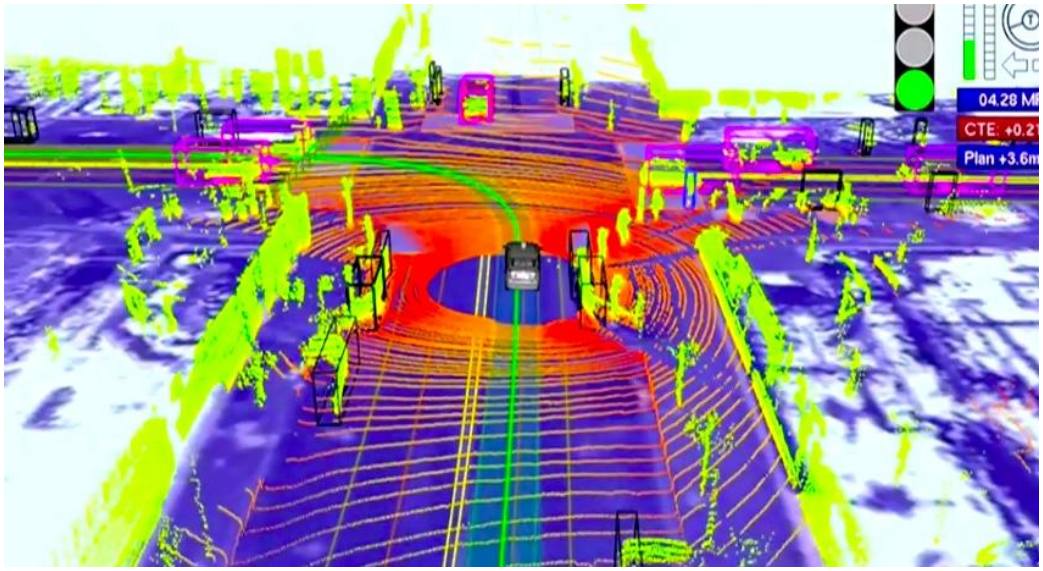
2.2.7 Autonomní vozidlo Google

Společnost Google v současné době vyvíjí plně autonomní vozidlo schopné pohybu v běžném provozu kompletně autonomně. Cílem je vytvořit vozidlo, které sníží nehodovost, zvýší pohodlí cestování a navíc bude šetrnější k životnímu prostředí. Google postavil takovýchto vozidel několik pro simultánní cestování a v roce 2012 měla tato vozidla najezděno již přes 300 000 km s pouze jedinou nehodou, při které podle vyjádření společnosti bylo vozidlo řízeno člověkem. [32]



Obr. 11 Snímek z prezentace zobrazující jednotlivé senzory [33]

Celý systém pracuje na bázi sbírání přibližně 1TB dat za vteřinu z několika senzorů pro kompletní přehled o okolí. Základem systému snímání okolí je LIDAR značky Velodyne umístěný na střeše, který skenuje okolí vozu pomocí 64 laserových paprsků. Z těchto dat je vytvořena detailní 3D mapa okolí, která je následně zkombinována s mapami s vysokým rozlišením již dříve získanými společností Google pomocí aut vyslaných před tímto vozidlem. Díky těmto mapám je vozidlo schopno odlišit chodce a jiné pohybující se objekty od pevných objektů. V náraznících jsou umístěny 4 radary pro detekci vzdálených objektů. Za zpětným zrcátkem je připevněna kamera pro přesnější detekci pohybujících se objektů a určení signálu semaforů. Auto je navíc vybaveno systémem pro určení polohy složeného z GPS, snímání pohybu kol a inerciální měřící jednotky. [34]



Obr. 12 Svět z pohledu Google auta [34]

Na Obr. 12 lze vidět celkový pohled tohoto autonomního vozidla na svět při odbočení vlevo na křižovatce. Je zde růžové označený okolní provoz důležitý pro rozhodování situace na křižovatce. Černě je označen provoz pro toto rozhodování nedůležitý. Celý systém je schopen zvládnout 99% všech situací vzniklých v silničním provozu. Situace, které tento systém zatím nezvládá, jsou například zasněžená vozovka nebo nepředvídatelný pohyb ostatních účastníků provozu, například průjezd jiného vozu na červenou. [33][34]

V případě rozšíření tohoto systému společnost Google slibuje ušetření části nákladů na palivo díky lepšímu využití mezer v provozu, jednoduššího sdílení či pronájmu aut a tím změnu pohledu na automobilový svět. Výhledově chce tento projekt vytvořit síť těchto autonomních vozidel schopných komunikace mezi sebou za účelem vytváření konvojů a sdílení dat o provozu. A hlavně je zde snaha o výrazné snížení nehodovosti celosvětového silničního provozu. [33]

3 VYBRANÉ ŘEŠENÍ ŘÍZENÍ

Abych mohl vybrat způsob řízení nejlépe se hodící k požadovanému konvoji, musím vzít v úvahu omezení co se týče rozměru v prostoru i náročnost na výpočet. V následující kapitole detailně popíši použitý hardware, na kterém jsem měl toto řešení realizovat. Na konci této kapitoly vyberu nejlepší metodu autonomního řízení vhodnou k řešení tohoto problému a tento výběr odůvodním.

Vybraná metoda řízení měla být experimentálně otestována na konvoji sestávajícím ze 2 RC modelů v měřítku 1:10. Konvoj měl být realizován s prvním vozidlem řídícím a druhým řízeným. Řízené vozidlo bylo osazeno jednodeskovým počítačem BeagleBoard xM pro zpracování veškerých dat ze senzorů a následně předání pokynů jednodeskovému systému řízení serv, který se stará o řízení pohybu vozidla. Jak jsem již napsal v úvodu, vybrané řešení řízení musí pokrýt senzorickou část včetně softwaru zpracování dat na BeagleBoardu.

3.1 RC model

Jako testovací RC model byl vybrán HIMOTO Monster Truck EMXT-1 4x4 RTR v měřítku 1:10. Tento model má rozměry 410 mm na délku a 300 mm na šířku, rozvor kol je 275 mm s výškou podvozku 32 mm. Je osazen koly s průměrem 118 mm a šířkou 52 mm. [35]

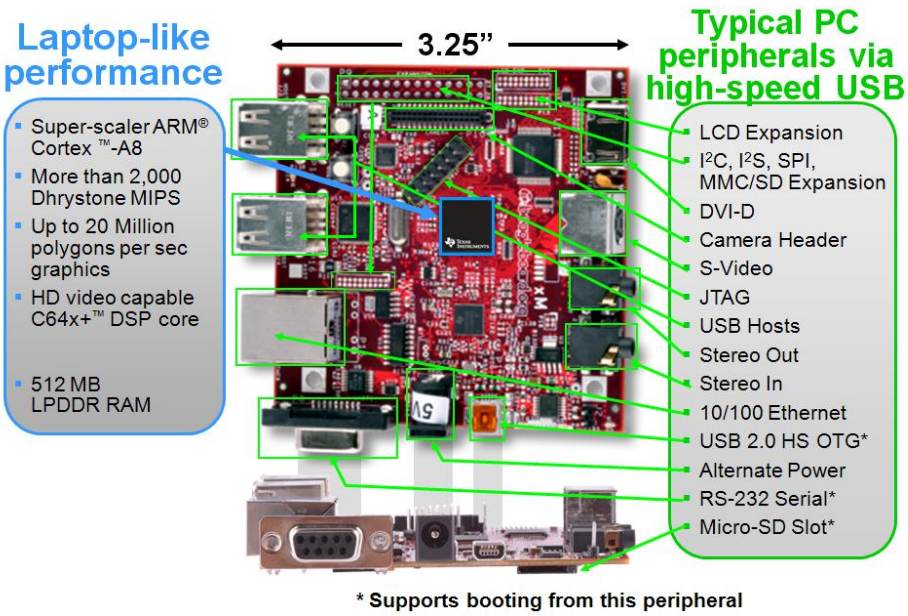
Obr. 13RC model HIMOTO EMXT-1 [35]

Podvozek tohoto RC modelu je standartní podvozek dodávaný firmou HIMOTO pro modely v měřítku 1:10. Je plně odpružený s použitím 8 kusů olejových tlumičů, u kterých je nastavitelná tuhost. Všechny převody jsou kompletně zakrytované, tudíž se do nich případně z nich nemůže dostat žádný cizí předmět. Celý podvozek je zpevněn 3 mm hliníkovou deskou, která spojuje přední a zadní diferenciál. [35]

Základem elektroniky celého modelu je elektronický regulátor HSP SP-03018, který zvládá trvalé proudové zatížení 30 A a špičkové až 300 A. O odvod tepla se stará pasivní chladič připevněný k tomuto regulátoru. Tento regulátor plynule reguluje pohon ve 3 funkcích: plyn, brzda a zpátečka. Jako napájení podporuje baterie Ni-Cd/Ni-Mh o napětí 7,2 V. K pohonu je zde stejnosměrný motor RC540 s maximálními otáčkami až 20 000 ot/min a s maximální rychlostí 35 km/h. Výkon od motoru je přenášen přes ocelový kardan na všechna 4 kola. Celý model má proporcionální řízení a obsahuje 4 kanálový 2,4 GHz přijímač a je ovládán pomocí dvoukanálové 2,4 GHz volantové RC soupravy.[35]

3.2 BeagleBoard xM Rev.C

Pro vlastní zpracování výstupu senzorů byl použit jednodeskový počítač BeagleBoard xM s označením C. Termín jednodeskový počítač znamená, že místo použití základní desky a různých komponentů má tento počítač vše na jedné desce osazené jedním či více mikroprocesory, pamětí a vstupy a výstupy. Jejich použití je zejména pro výzkumné a embedded systémy. Beagleboard xM je osazen procesorem Cortex A8 s frekvencí 1 GHz od firmy Texas Instruments, operační pamětí Micron POP 512 MB MDDR SDRAM pracující na frekvenci 200 MHz. Celkové rozměry tohoto jednodeskového počítače jsou 8,3×8,3 cm. Na Obr. 14 Obr. 14 Vyznačené rysy BeagleBoardu xMjsou vyznačené hlavní rysy BeagleBoardu xM včetně všech vstupů a výstupů. [36]



Obr. 14 Vyznačené rysy BeagleBoardu xM [37]

BeagleBoard xM má mnoho možných využití vzhledem k tomu, že je postaven jako platforma pro vývoj otevřených aplikací. Dle výrobce ho lze použít například pro poskytování webových služeb, jako domácí media center, pro vývoj software všemožných typů od firmware až po grafické aplikace a samozřejmě robotiku. Na tento jednodeskový počítač lze nahrát jakýkoliv operační systém určený pro tento typ architektury ARM. [38]

3.3 Vybraná metoda řízení

Vzhledem k nutnosti malé prostorové i výpočetní náročnosti jsem musel vybrat metodu s co nejmenším počtem senzorů případně co nejmenší senzory schopné zjistit nejenom rychlost nebo vzdálenost od vozidla vepředu, ale také směr jeho pohybu. Tyto senzory musí mít malou spotřebu vzhledem k nutnosti napájet celý systém z baterie určené k provozu vozidla. Pro tento úkol je ideální řízení pomocí zpracování obrazu z malé kamery. BeagleBoard xM Rev.C je schopný vývoje 3D aplikací a současně rozměry jednoduchých USB kamer jsou v řádech centimetrů, tudíž se bez problémů vejdou na příslušný RC model.

Nyní je nutno vyřešit softwarové řešení. Detekci pohybu z obrazu lze realizovat několika způsoby. Prvním z nich je optic flow, tedy vypočítání pohybu pomocí rozdílů ve 2 po sobě jdoucích snímcích. Toto má nevýhodu jednak obtížného naprogramování, ale také náročnosti na výpočet. Takže jsem volil metodu jinou, optickou triangulaci. Metoda optické triangulace je schopna jednoduše zjistit vzdálenost od vozidla pomocí nalezení 2 rysů tohoto vozidla a znalosti jejich vzdálenosti od sebe při definované vzájemné vzdálenosti vozidel. Tímto způsobem lze určit vzdálenost od vozidla a následně tuto vzdálenost držet pomocí nastavování rychlosti vlastního vozidla. Pro určování směru pohybu jsem vzal v potaz natočení vedoucího RC modelu vůči modelu řízenému. Tudíž kdybych byl schopen identifikovat 3 rysy a znal jejich vzájemné vzdálenosti, tak při natočení vedoucího RC modelu se tyto

poměry změní a tím pádem mohu vypočítat úhel mezi směry pohybu jednotlivých modelů a natáčením kol tento úhel držet na 0°.

Nakonec je zde problém rysů, které potřebuji identifikovat. Tento RC model není nijak barevně kontrastní vůči okolí a nemá žádné jednoduše rozeznatelné rysy. Takže jsem musel tyto rysy přidat. Tyto rysy musí být jednoduše identifikovatelné z různých úhlů, což splňuje identifikace obvodu míčku v prostoru, tedy nalezení kruhu. Navíc pro snadnou identifikaci zde musí být maximální kontrast při různém osvětlení. Nejjednodušším řešením tohoto problému je použití matné černé desky, na které jsou připevněny 3 pingpongové míčky.

4 INSTALACE SOFTWARE NA BEAGLEBOARD XM

Abychom mohli toto řešení experimentálně ověřit na daném konvoji RC modelů, je potřeba na BeagleBoard xM nainstalovat operační systém a další software. V této kapitole tedy popíši postup instalace celého systému a dalšího software potřebného k tomuto ověření. Celá instalace je problematická co se týče neshod verzí a chybějících balíčků kvůli neustálému vývoji tímto směrem. Proto jsem psal postup velmi systematicky, jeho dodržení eliminuje většinu chyb.

4.1 Instalace Ubuntu

Na microSD kartě dodávané spolu s BeagleBoardem xM jsou všechny potřebné soubory pro spuštění validačního obrazu systému Angstrom. [36] Tento systém zde slouží pro ověření funkčnosti celé desky, ale vzhledem k jeho malému rozšíření a tudíž i menší podpoře jsem zvolil jiný systém. Vybral jsem jeden z nejpoužívanějších unixových operačních systémů systému, Ubuntu. Výhoda tohoto systému je v mnoha jeho použití velkého počtu jeho verzí na jednodeskovém počítači BeagleBoard xM. Díky této větší uživatelské základně je podpora BeagleBoardu xM vyšší mezi uživateli a tím je i snadnější řešení problémů, které se mohou vyskytnout. Celý popsaný postup instalace jádra je proveden na jiném počítači s operačním systémem Ubuntu 12.10 Quantal Quetzal. Toto lze provést i pod systémem Windows se softwarem schopným emulace systému Ubuntu, ale celý proces by trval mnohem déle. Proto doporučuji vlastní instalaci celého systému Ubuntu, veškeré informace a potřebné postupy lze najít na českých stránkách Ubuntu. [39][40]

4.1.1 Instalace demo jádra

Nejprve pro ověření funkčnosti Ubuntu na BeagleBoardu xM doporučuji stáhnout demoverzi tohoto systému. Tato demoverze nemá plnou podporu periferií. Může posloužit k rychlé a jednoduché demonstraci systému Ubuntu na BeagleBoardu xM, ale také otestování vlastní schopnosti instalace jiného systému na BeagleBoard xM. [40]

Nejprve musíme stáhnout demo obraz.

```
#1  wget http://rcn-ee.net/deb/rootfs/quantal/ubuntu-12.10-console-armhf-2013-04-26.tar.xz
```

Obr. 15 Stažení demo image [40]

Následně ověříme správnost stažení pomocí CRC.

```
#1  md5sum ubuntu-12.10-console-armhf-2013-04-26.tar.xz
#2  b7cd2f94c0dbc75f8d6897e559d52c0a  ubuntu-12.10-console-armhf-2013-04-26.tar.xz
```

Obr. 16 Ověření správnosti stažení [40]

Poté tento obraz rozbalíme a přejdeme do složky s obrazem.

```
#1  tar xJf ubuntu-12.10-console-armhf-2013-04-26.tar.xz
#2  cd ubuntu-12.10-console-armhf-2013-04-26
```

Obr. 17 Rozbalení demo obrazu [40]

Po rozbalení vsuneme SD kartu a zjistíme její umístění.

```
#1  sudo ./setup_sdcard.sh --probe-mmc
```

Obr. 18 Zjištění umístění SD karty [40]

Napsáním tohoto příkazu se nám zobrazí něco podobného jako na Obr. 19. Je zde vyznačeno zobrazené umístění SD karty.

```
theta@hloupej:~/beaglePREINST/ubuntu-12.10-console-armhf-2013-03-28$ sudo ./setu
p_sdcard.sh --probe-mmc

Are you sure? I Don't see [/dev/idontknow], here is what I do see...

/sbin/fdisk -l:
Disk /dev/sda: 320.1 GB, 320072933376 bytes
Disk /dev/sdb: 7985 MB, 7985954816 bytes

mount:
/dev/sda5 on / type ext4 (rw,errors=remount-ro)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
```

Obr. 19 Výpis umístění SD karty v konzoli

Nyní nainstalujeme samotný obraz na SD kartu. Pokud je umístění SD karty jiné, než /dev/sdb, pak ho změníme v příkazu na Obr. 20. Skript se nás znovu zeptá na výběr umístění SD karty (Obr. 21), které pokud je správné, potvrdíme.

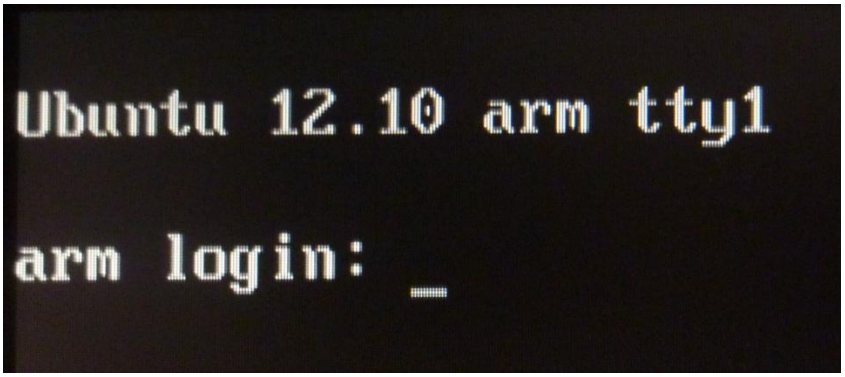
```
#1  sudo ./setup_sdcard.sh --mmc /dev/sdb --uboot beagle_xm
```

Obr. 20 Instalace obrazu na SD kartu [40]

```
theta@hloupej:~/beaglePREINST/ubuntu-12.10-console-armhf-2013-03-28$ sudo ./setu
p_sdcard.sh --mmc /dev/sdb --uboot beagle_xm

I see...
/sbin/fdisk -l:
Disk /dev/sda: 320.1 GB, 320072933376 bytes
Disk /dev/sdb: 7985 MB, 7985954816 bytes

lsblk:
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda      8:0    0 298.1G  0 disk
├─sda1   8:1    0   100M  0 part
├─sda2   8:2    0 195.2G  0 part
├─sda3   8:3    0     1K  0 part
├─sda5   8:5    0 102.3G  0 part /
└─sda6   8:6    0   476M  0 part [SWAP]
sdb      8:16   1   7.4G  0 disk

Are you 100% sure, on selecting [/dev/sdb] (y/n)? y
```

Obr. 21 Potvrzení výběru umístění SD karty

Kartu vložíme do BeagleBoardu xM. Pro tento test nám stačí připojit klávesnici a monitor před HDMI kabel. Pozor, pokud je BeagleBoard xM zapnut, nesmí se manipulovat s HDMI kabelem. Pokud chceme například změnit monitor, musíme celý počítač vypnout, přepojit kabel, a poté znovu zapnout. Po spuštění BeagleBoardu xM začne načítat celý systém a po pár vteřinách se zobrazí požadavek na přihlášení (Obr. 22). Pro přihlášení použijeme defaultní údaje na Obr. 23.



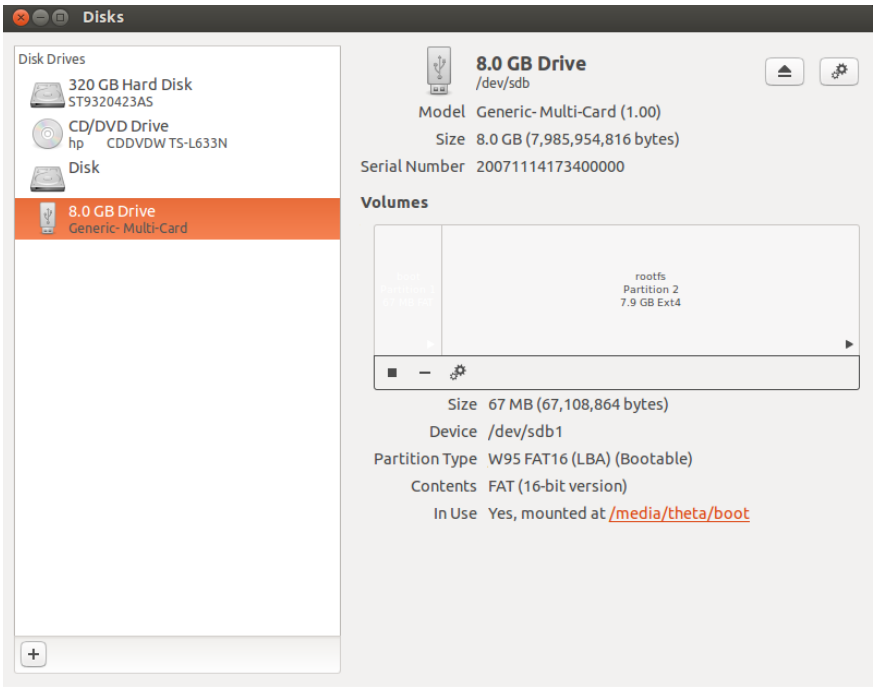
Obr. 22 Přihlášení do systému s demo obrazem

```
#1  Login: ubuntu
#2  Heslo: temppwd
```

Obr. 23 Přihlašovací údaje

4.1.2 Kompilace nového jádra s SGX

Pokud instalace demo jádra proběhla bez problémů a podařilo se nám přihlásit, můžeme přejít ke kompilaci vlastního jádra včetně grafické akcelerace pomocí modulů SGX. Doporučuji SD kartu nejdříve naformátovat. Pod Ubuntu je pro tuto úlohu grafický správce disků Disks (Obr. 24).



Obr. 24 Formátování v programu Disks

Nové jádro budeme instalovat pomocí systému správy verzí git. Ten je dostupný pomocí balíčkovacího systému APT. Nejprve tedy updatujeme databázi tohoto systému, abychom dostali nejnovější verzi, a poté stáhneme program git.

```
#1  sudo apt-get update
#2  sudo apt-get install git
```

Obr. 25 Instalace programu git

Po úspěšné instalaci programu git můžeme přikročit ke krokům kompilace nového jádra s moduly SGX. Doporučuji vytvoření nové složky, ve které budeme poté pracovat. Nazvěme ji například beagle.

```
#1  mkdir beagle
#2  cd beagle
```

Obr. 26 Vytvoření nového adresáře pro práci

Do této složky stáhneme vše potřebné pomocí programu git příkazem na Obr. 27.

```
#1  git clone git://github.com/RobertCNelson/stable-kernel.git
```

Obr. 27 Stažení potřebných souborů [40]

Následně se přesuneme do adresáře s kernelem a zvolíme vhodnou verzi jádra. Postupem času

zde přibývají novější verze, tudíž doporučuji volit tu nejnovější, pokud s ní nebudou problémy.

```
#1  cd stable-kernel
#2  git checkout origin/v3.9.x -b v3.9.x
```

Obr. 28 Přesun ke kernelu a nastavení větve

Spustíme příkaz pro zkompilování kernelu.

```
#1  ./build_kernel.sh
```

Obr. 29 Kompilace kernelu [40]

Následně zkompilujeme SGX moduly. Toto chvíli potrvá, protože je nejdříve musíme stáhnout.

```
#1  ./sgx_build_modules.sh
```

Obr. 30 Kompilace modulů SGX [40]

Před samotným nainstalováním na SD kartu je potřeba změnit cestu k SD kartě v system.sh. Otevřeme si tento soubor (Obr. 31) a najdeme řádek s MMC=/dev/sde a změníme ho podle umístění karty. V našem případě na MMC=/dev/sdb.

```
#1  gedit system.sh
```

Obr. 31 Úprava system.sh

Poté nainstalujeme systém na SD kartu.

```
#1  ./tools/install_kernel.sh
```

Obr. 32 Příkaz pro instalaci systému na SD kartu

Nyní je nainstalovaný kernel a zkompilovaná grafická akcelerace SGX. Poté je potřeba na SD kartu zkopírovat moduly SGX pro následné nainstalování na BeagleBoard xM. Samozřejmě v případě jiné verze bude název souboru odpovídat té verzi. Nejjednodušší způsob je tyto soubory zkopírovat graficky, ale na to je potřeba oprávnění správce. To lze získat spuštěním správce souborů jako správce (Obr. 33 Spuštění správce souborů nautilus jako root).

```
#1  sudo nautilus
```

Obr. 33 Spuštění správce souborů nautilus jako root

Nyní najdeme soubory s moduly SGX (Obr. 34 Soubory s moduly SGX) a zkopírujeme je do /opt/sgx na SD kartě.

```
#1  GFX_4_08_00_01_libs.tar.gz
#2  GFX_Linux_SDK.tar.gz
```

Obr. 34 Soubory s moduly SGX

Vsuneme SD kartu do BeagleBoardu xM. Po nastartování BeagleBoardu xM se nám zobrazí dotaz na přihlášení. Znovu použijeme defaultní údaje (Obr. 23) a přihlásíme se do systému. Poté nainstalujeme grafickou akceleraci SGX.

```
#1  cd /opt/sgx
#2  sudo tar xf GFX_4.08_00_01_libs.tar.gz
#3  sudo ./install-sgx.sh
```

Obr. 35 Instalace SGX na zařízení

Restartujeme zařízení (Obr. 36) a otestujeme funkčnost modulů SGX (Obr. 37). Tento test by měl při úspěchu vypsat u každého kroku OK.

```
#1  sudo reboot
```

Obr. 36 Příkaz pro restart Ubuntu

```
#1  cd /usr/bin/armhf/es5.0
#2  ./sgx_blit_test
```

Obr. 37 Test modulů SGX [40]

4.1.3 Instalace prostředí

Před dalším postupem a instalováním nových programů doporučuji pro jistotu osvěžit databázi balíčků APT a stáhnout nejnovější verze programů. Tento krok je formální, protože při použití nejnovějšího kernelu by měly být všechny nainstalované programy na nejnovější verzi.

```
#1  sudo apt-get update
#2  sudo apt-get upgrade
```

Obr. 38 Upgrade pomocí správce balíčků APT

Nyní nainstalujeme desktopové prostředí. Vzhledem k výkonu BeagleBoardu xM doporučuji vybrat některé z mnoha lighweight prostředí, tedy prostředích minimálně náročných na hardware. Já jsem vybral prostředí LXDE (Lightweight X11 Desktop Enviroment). Toto prostředí je velmi blízké všem uživatelům Windows Vista nebo Windows 7 svým vzhledem, protože je odvozeno od prostředí KDE. Najdeme zde známý panel aplikací u spodní části obrazovky i s panelem podobným panelu Start v jeho levé části. Toto prostředí nainstalujeme pomocí příkazu (Obr. 39), poté systém restartujeme (Obr. 36).

```
#1  sudo apt-get install lxde
```

Obr. 39 Instalace LXDE

Po restartu by nám systém měl naběhnout do desktopového prostředí LXDE. Zde se přihlásíme jako „Demo user (Ubuntu)“ s heslem „temppwd“. Tím se dostáváme do samotného prostředí LXDE.

Obr. 40 Přihlášení do LXDE

Obr. 41 Plocha LXDE po prvním přihlášení

4.2 Instalace Open CV

K tomuto experimentálnímu ověření vybraného řešení jsem použil knihovny OpenCV, tudíž abych ho mohl otestovat přímo na BeagleBoardu xM, musím tam tyto knihovny nainstalovat. Nejprve je potřeba nainstalovat celkem dlouhý seznam prerekvizit. Pokud příkaz oznámí u kteréhokoliv tohoto programu chybu, patrně se již daný balíček nevyskytuje na serveru a musíme najít alternativu. Většinou je alternativou novější verze daného balíčku, někdy je tento balíček s novou verzí přejmenován. Důležité je při hledání balíčků dodržet architekturu BeagleBoardu xM – armhf, protože často se stává, že daný balíček existuje, ale ne pro architekturu armhf. Na Obr. 42 je seznam příkazů

pro nainstalování veškerých prerekvizit nutných k instalaci OpenCV na BeagleBoardu xM. Je to upravený seznam z [41] pro výše uvedené verze kernelu.

```
#1  sudo apt-get install build-essential
#2  sudo apt-get install cmake cmake-gui
#3  sudo apt-get install pkg-config
#4  sudo apt-get install libpng++-dev libpng3
#5  sudo apt-get install libpnglite-dev libpngwriter0-dev libpngwriter0c2
#6  sudo apt-get install zlib1g-dbg zlib1g zlib1g-dev
#7  sudo apt-get install libjasper-dev libjasper-runtime libjasper1
#8  sudo apt-get install pngtools libtiff4-dev libtiff4 libtiffxx0c2 libtiff-tools
#9  sudo apt-get install libjpeg8 libjpeg8-dev libjpeg8-dbg libjpeg-progs
#10 sudo apt-get install ffmpeg libavcodec-dev libavcodec53 libavformat53 libavformat-dev
#11 sudo apt-get install libgstreamer0.10-0-dbg libgstreamer0.10-0  libgstreamer0.10-dev
#12 sudo apt-get install libxine1-ffmpeg  libxine-dev libxine1-bin
#13 sudo apt-get install libunicap2 libunicap2-dev
#14 sudo apt-get install libdc1394-22-dev libdc1394-22 libdc1394-utils
#15 sudo apt-get install swig
#16 sudo apt-get install libv4l-0 libv4l-dev
#17 sudo apt-get install python-numpy
#18 sudo apt-get install libpython2.7 python-dev python2.7-dev
#19 sudo apt-get install libjpeg-progs libjpeg-dev
```

Obr. 42 Příkazy na instalování prerekvizit OpenCV

Následně vytvoříme složku pro OpenCV a stáhneme ho pomocí gitu.

```
#1  mkdir opencv
#2  cd opencv
#3  git clone git://code.opencv.org/opencv.git
```

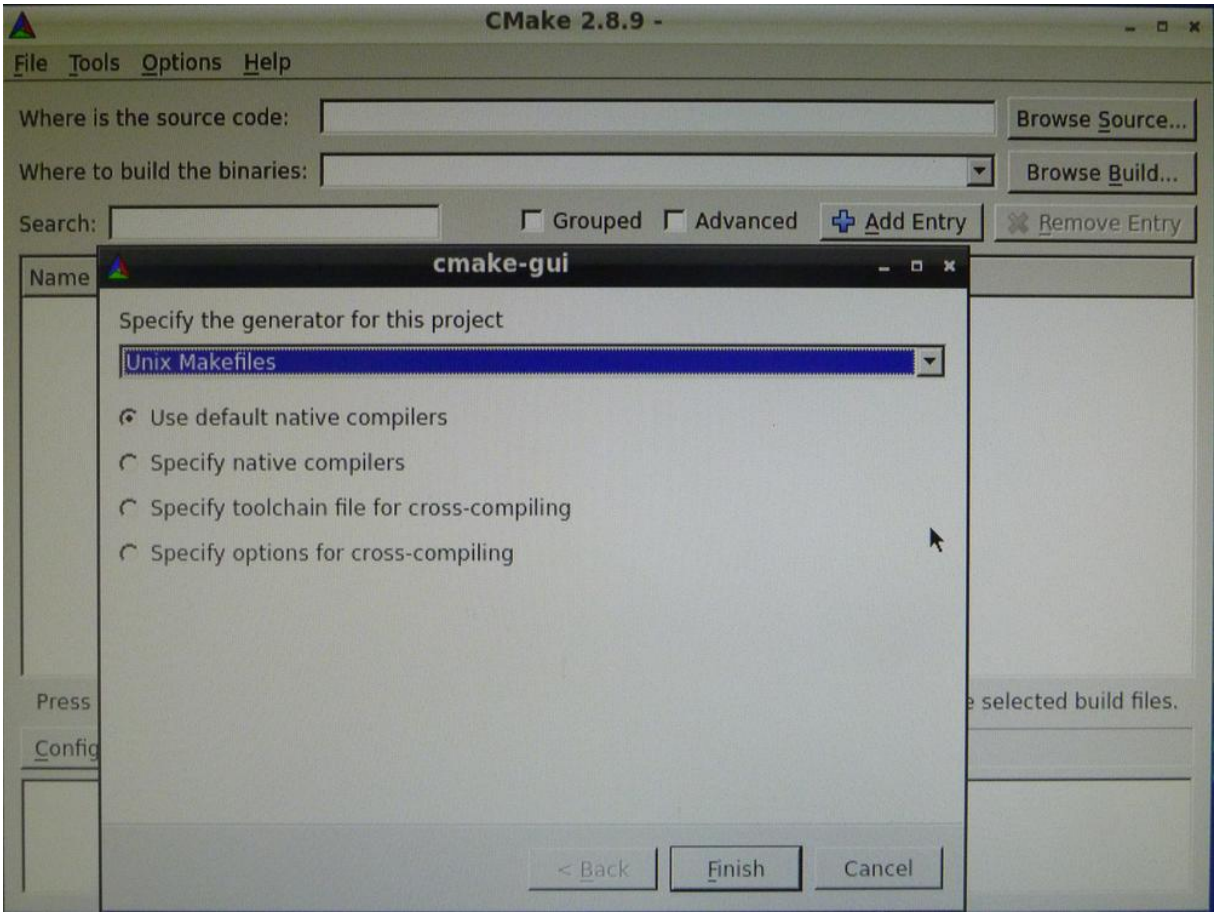
Obr. 43 Stažení OpenCV do vytvořeného adresáře

Poté spustíme grafickou verzi cmake s právy roota pro snadnější nastavování.

```
#1  sudo cmake-gui
```

Obr. 44 Spuštění GUI programu cmake [41]

Zde nejdříve nastavíme cestu k souborům stažených gitem na /home/ubuntu/opencv/ a cestu pro kompilaci na složku pro make, klidně znovu /home/ubuntu/opencv/. Následně v menu tools –> configure změníme generátor na Unix Makefiles. Po odkliknutí tlačítka „Finish“ se nám vygeneruje konfigurační soubor pro kompilátor cmake.



Obr. 45 Nastavení generátoru konfiguračního souboru

Nyní tento kompilátor spustíme a nainstalujeme zkompilované soubory příkazem na Obr. 46. Tímto krokem jsme úspěšně nainstalovali OpenCV na BeagleBoard xM.

```
#1  make;make install
```

Obr. 46 Příkaz pro spuštění kompilace [41]

5 OVĚŘENÍ VYBRANÉHO ŘEŠENÍ

Vybrané řešení jsem ověřoval 2 způsoby, v C++ s využitím knihoven OpenCV a v C# s knihovnami Emgu CV. Vzal jsem v úvahu své větší zkušenosti s jazykem C# a proto jsem začal tímto způsobem. Pro oba způsoby jsem použil funkci Houghovy kruhové transformace, která je zabudována v obou těchto knihovnách.

5.1 C# s využitím Emgu CV

Emgu CV je pouze wrapper knihoven OpenCV, tedy knihoven pro zpracování obrazu, což umožňuje volat funkce OpenCV jazykem .NET, tudíž i jazykem C#. Dle oficiálních stránek je Emgu CV použitelné napříč platformami a operačními systémy, pokud se zkompiluje v kompilátoru Mono. [42]

Pro ověření schopnosti navigace pomocí detekce kruhů jsem postupoval podle návodu [43] na oficiálních stránkách Emgu CV. Použil jsem volně dostupné části kódu nutné pro detekci kruhů s vlastním nastavení Houghovy kruhové transformace podle Obr. 47. K tomuto nastavení jsem se dostal experimentální cestou a je dostačující pro běžné denní nebo kancelářské osvětlení.

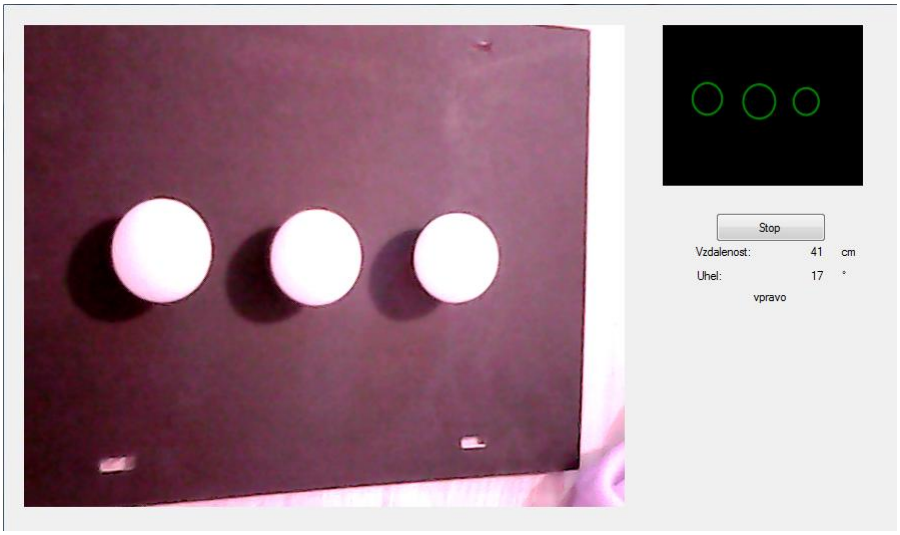
```
CircleF[] circles = gray.HoughCircles(
        cannyThreshold,
        circleAccumulatorThreshold,
        1.9, //Resolution of the accumulator used to detect centers of the circles
        50.0, //min distance
        10, //min radius
        80 //max radius
        )[0]; //Get the circles from the first channel
```

Obr. 47 Nastavení HoughCircles()

Následně jsem musel vytvořit funkci pro určení úhlu a vzdálenosti. Její vstupy jsem vzal středy 3 prvních nalezených kruhů na reálných míčcích. S tímto nastavením parametrů najde funkce vždy maximálně 1 kruh na každý míček a s černým matným pozadím je snížena pravděpodobnost nalezení dalších kruhů. Tyto souřadnice jsem si nejprve seřadil podle osy X, abych měl jistotu výpočtu ze stále stejných míčků. Následně jsem tyto body spojil a změřil délku této úsečky. Poté jsem vypočítal vzdálenost předpokladem znalosti reálné vzdálenosti a převrácením hodnoty změřené, neboť čím blíže jsem k míčkům, tím více pixelů jsou jejich středy od sebe. Následně jsem vypočítal úhel svíraný vozidly změřením poměrů jednotlivých úseček. Zde jsem musel kalibrovat velikosti úseček z důvodu nepřesné konstrukce, aby výsledný poměr při 90° byl 1. Na Obr. 48 je celý početní kód pro určení úhlu a vzdálenosti. A na Obr. 49 je celá API programu C# při testování této metody.

```
private void UrciUhelAVzdalenost(PointF bod1, PointF bod2, PointF bod3)
{

    if (bod1.X > bod2.X) { pointTMP = bod1; bod1 = bod2; bod2 = pointTMP; }
    if (bod2.X > bod3.X) { pointTMP = bod2; bod2 = bod3; bod3 = pointTMP; }
    if (bod1.X > bod2.X) { pointTMP = bod1; bod1 = bod2; bod2 = pointTMP; }

    if ((bod1 != PointF.Empty) && (bod2 != PointF.Empty) && (bod3 != PointF.Empty))
        cara12 = new LineSegment2DF(bod1, bod2);
        cara23 = new LineSegment2DF(bod2, bod3);

    if ((bod1 != PointF.Empty) && (bod2 != PointF.Empty) && (bod3 != PointF.Empty))
    {
        vzdalenost = ((340*40) / (cara12.Length + cara23.Length));
        uhel = ((90-(cara12.Length/(cara23.Length*1.05)*90))*2);
    }
}
```

Obr. 48 Kód pro určení úhlu a vzdálenosti



Obr. 49 Program C# v chodu

5.2 C++ s využitím Open CV

Následně jsem celý kód přepsal do C++ kvůli jednodušší implementaci pod Ubuntu. Postupoval jsem podle oficiální dokumentace Open CV [44] s cílem přesně replikovat C# kód. Bohužel, toto nebylo možné, protože funkce Houghovy kruhové transformace zde má jiné parametry, jak lze vidět na Obr. 50. Pokusil jsem se nastavit hodnoty podobné předchozímu případu (Obr. 47), ale s těmito hodnotami funkce nenašla ani jeden z míčků. Proto jsem experimentální cestou nastavil hodnoty (Obr. 50). S těmito hodnotami funkce nachází všechny míčky přibližně 80% času. Zbylých 20% času najde 0-2 míčků. Z tohoto důvodu jsem vypočítal pouze vzdálenost pro demonstrativní účely.

```
vector<Vec3f> circles;
HoughCircles(gray,                  //vstupní frame
             circles,               //seznam kruhů
             CV_HOUGH_GRADIENT,     //metoda detekce
             2,                     //prevracena hodnota rozliseni
             50,                    //minimalni vydalenost mezi stredy
             200,                   //vrchni threshold pro detektor hran Canny
             90);                   //threshold pro detekci stredu
```

Obr. 50 Parametry Houghovy transformace v Open CV

Výstup z funkce HoughCircles() je matice vektorů (X, Y, R). Tudíž například circles[2][0] značí souřadnici na ose X 3. kruhu. Vzdálenost jsem pro větší přesnost počítal jako přeponu vzdálenosti souřadnic X a Y.

```
if(circles.size() >= 2)
{
    if(circles.size() == 3)
    {
        vzdalenost = sqrt((circles[2][0]-circles[0][0])*(circles[2][0]-circles[0][0]) + (circles[2][1]-circles[0][1])*(circles[2][1]-circles[0][1]));
        vzdalenost = (10/vzdalenost)*310;
    }
    else if(circles.size() == 2)
    {
        vzdalenost = sqrt((circles[1][0]-circles[0][0])*(circles[1][0]-circles[0][0]) + (circles[1][1]-circles[0][1])*(circles[1][1]-circles[0][1]));
        vzdalenost = (10/vzdalenost)*620;
    }

    char text[20];
    sprintf(text,"%f",vzdalenost);

    putText(frame, text, cvPoint(30,30), FONT_HERSHEY_COMPLEX_SMALL, 0.8, cvScalar(200,200,250), 1, CV_AA);
}
```

Obr. 51 Kód pro výpočet vzdálenosti od míčků v C++

Samotná testovací aplikace je na Obr. 52. Zrovna v tomto případě funkce našla jen 2 míčky, ze kterých určila vzdálenost 42,4cm.

Obr. 52 Testovací aplikace v C++

6 ZÁVĚR

Zadáním mé bakalářské práce bylo vypracování rešeršní studie zabývající se současnými metodami řízení vozidla v konvoji a na základě této rešerše vybrat nejvhodnější metodu pro experimentální otestování na konvoji RC modelů v měřítku 1:10.

Implemetace jazyka C# pod Ubuntu, které není nativním prostředím pro tento jazyk, je problematické vzledem k nutnosti řešení netypických závislostí na instalovaných softwarových modulech. Tomuto problému je věnována kapitola 5.1. Jazyk C# je použit jako hlavní implementační nástroj pro vypočtení vzdálenosti a úhlu z obrazu snímaného kamerou. Tato funkce byla úspěšně realizována a experimentálně ověřena za použití počítače a externí kamery s rozlišením 640×480 px. Program byl limitován rozlišením kamery ale i tak bez problémů nacházel všechny 3 míčky ve vzdálenosti od 30 cm do 120 cm a v úhlu +- 45°.

Následně jsem nainstaloval operační systém Ubuntu na testovacím jednodeskovém počítači BeagleBoard xM. Poté jsem nainstaloval desktopové prostředí LXDE a knihovny pro zpracování obrazu Open CV. Pro implementaci C# kódu pod Ubuntu jsem se pokusil na BeagleBoard xM nainstalovat kompilátor Mono nutný pro kompilaci C# kódu a knihoven Emgu CV. Vzhledem k nedostatečné podpoře Mono na BeagleBoardu xM nebyla instalace zcela bezproblémová.

Pro ověření jiné možnosti přípravy experimentálního software byla zvolena metoda uprávy existujícího kódu v C# do nativního programovacího jazyka C++, jehož implementace pod systémem Ubuntu je zcela bezproblémová a pro který jsem již na BeagleBoard xM nainstaloval veškeré nutné programy. Vzhledem k tomu, že implementace knihovny OpenCV se v jednodlivých prostředích mírně liší, nepovedlo se nastavit parametry funkce Houghovy kruhové transformace dostatečně přesně pro nalezení všech míčků. Funkce implementovaná v C++ nalezla míčky pouze v přibližně 80% případů, což je nedostatečné pro nasazení na reálném zařízení.

Knihovna OpenCV tvoří základ většiny moderních robotických systémů využívajících zpracování obrazu, proto je tato knihovna neustále vyvíjena. Provedené experimenty budou sloužit jako základ pro další práce s podobnou tématikou.

7 SEZNAM POUŽITÉ LITERATURY

[1] Sebastian Thrun, Wolfram Burgarn, and Dieter Fox. Probabilistic Robotics (Inteligent Robotics and Autonomous Agent series). *Inteligent robotics and autonomous agents.* The MIT Press, August 2005.

[2] Howie Choset, Kevin M. Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia E. Kavraki, and Sebastian Thrun. *Principles of Robot Motion: theory, Algorithms, and Implementations (Inteligent Robotics and Autonomous Agents)*. The MIT Press, June 2005.

[3] Toyota Motor Czech. *Technologie* [online]. [cit.10.5.2013]. Dostupné z:<http://www.toyota.cz/cars/new_cars/toyota_tech/acc.tmex>

[4] Škoda. *Adaptivní tempomat* [online]. [cit.10.05.2013]. Dostupné z:<http://new.skoda-auto.com/cs/models/HotspotDetail?HotspotName=C40%20-%20Adaptive%20Cruise%20Assistant%20%5BA7%5D&WebID=a03c2c80-b27c-4cea-9628-1b82435c5635&Page=technology>

[5] Wikimedia Foundation. *Autonomous cruise control system* [online]. 13.04.2013 [cit.10.05.2013]. Dostupné z:<https://en.wikipedia.org/wiki/Autonomous_cruise_control_system>

[6] Volvo. *Adaptivní tempomat* [online]. [cit.10.05.2013]. Dostupné z:<https://www.media.volvocars.com/media/images/low/18356_4_13.aspx?>

[7] AREM, Bart van, DRIEL, Cornelie J. G. van, VISSER, Ruben. The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristic. *IEEE transactions on inteligent transportation systems* [online]. 2006, č. 4, prosinec [cit.10.05.2013]. Dostupné z:<http://doc.utwente.nl/58157/1/Arem06impact.pdf>

[8] SARTRE-Consortium. *SARTRE Project* [online]. [cit.11.05.2013]. Dostupné z:<http://www.sartre-project.eu/en/Sidor/default.aspx>

[9] SARTRE-Consortium. *Platooned traffic can be integrated with other road users on conventional highways* [online]. 17.09.2012 [cit.11.05.2013]. Dostupné z:<http://www.sartre-project.eu/en/press/Documents/SARTRE%20final%20partner%20release.pdf>

[10] ERTICO. *CVIS* [online]. 2009 [cit.11.05.2013]. Dostupné z:<http://www.cvisproject.org/>

[11] European Commision. *CVIS* [online]. 28.02.2006 [cit.11.05.2013]. Dostupné z:<http://ec.europa.eu/information_society/activities/esafety/doc/esafety_2006/spectrum_28feb2006/5_cvis_spectrum_workshop_r2.pdf>

[12] VisLab. *The VisLab Intercontinental Autonomous Challenge* [online]. [cit.17.05.2013]. Dostupné z:<http://viac.vislab.it/>

[13] VisLab. *Konvoj v Samaře* [online]. [cit.17.05.2013]. Dostupné z:<http://viac.vislab.it/wp-content/themes/viac/galleries/my_galleries/2010-08-28-samara/120.jpg>

[14] Česká televize. *Stop – Mrtvý úhel* [online]. 2006 [cit.12.05.2013]. Dostupné z:<http://www.ceskatelevize.cz/porady/1096067152-stop/206562230400036/>

[15] KŘIVDA, Vladislav, ŠKVAIN, Václav. *Mrtvý úhel* [online]. 2011 [cit.12.05.2013]. Dostupné z:<http://kds.vsb.cz/mkk/img57.gif>

[16] Ford. *Blind Spot Information System (BLIS) with Cross-Traffic Alert* [online]. 2012, červenec

[cit.12.05.2013]. Dostupné z:<http://media.ford.com/images/10031/BLIS.pdf>

[17] Infinity Worldwide, *Enhanced confidence with Guiding technology* [online]. [cit.13.05.2013]. Dostupné z:<http://www.infinitiusa.com/now/technology/blind-spot-intervention-system.html>

[18] VISVIKIS, C., SMITH, T. L., PITCHER, M. SMITH, M. (TRL). *Study on lane departure warning and lane change assistant systems* [HTML document]. Transport Research Laboratory, November 2008. [cit.13.05.2013]. Dostupné z:<http://ec.europa.eu/enterprise/sectors/automotive/files/projects/report_ldwlca_en.pdf>

[19] Das Auto Magazine. *Lane Assist* [online]. [cit.13.05.2013]. Dostupné z:<http://www.dasauto-magazine.com/sites/default/files/styles/article_bg_2-1/public/size2_assistenz-bildgalerie-lane-assist.jpg>

[20] Wikimedia Foundation. *Intelligent Parking Assist System* [online]. 14.3.2012 [cit.16.05.2013]. Dostupné z:<http://en.wikipedia.org/wiki/Intelligent_Parking_Assist_System>

[21] AISIN SEIKI Co.,Ltd.. *New Intelligent Parking Assist Jointly Developed with Toyota Motor Corporation Installed on Lexus LS460* [online]. 29.06.2006 [cit.16.05.2013]. Dostupné z:<http://www.aisin.com/news/products/060929.html>

[22] Wikimedia Foundation. *Advanced Emergency Braking System* [online]. 22.03.2012 [cit.16.05.2013]. Dostupné z:<http://en.wikipedia.org/wiki/Advanced_Emergency_Braking_System>

[23] DeviceGuru. *Ford collision avoidance* [online]. [cit.16.05.2013]. Dostupné z:<http://deviceguru.com/files/ford-collision-avoidance.jpg>

[24] European Commison. *Impact Assessment* [online]. 23.05.2008[cit.16.05.2013]. Dostupné z:<http://ec.europa.eu/enterprise/sectors/automotive/files/safety/sec_2008_1908_en.pdf>

[25] Wikimedia Foundation. *DARPA Grand Challenge* [online]. 12.03.2012 [cit.17.05.2013]. Dostupné z:<http://en.wikipedia.org/wiki/DARPA_Grand_Challenge>

[26] HOOPER, Joseph. From Darpa Grand Challenge 2004DARPA's Debacle in the Desert. *Popular Science* [online]. 06.04.2004 [cit.17.05.2013]. Dostupné z:<http://www.popsci.com/scitech/article/2004-06/darpa-grand-challenge-2004darpas-debacle-desert>

[27] DARPA. *Grand Challenge '05* [online]. 31.12.2007 [cit.17.05.2013]. Dostupné z:<http://archive.darpa.mil/grandchallenge05/>

[28] 3rd St. Production Services, *Grand Challenge logo* [online]. [cit.17.05.2013]. Dostupné z:<http://www.3rd-st.com/Darpa/GrandChalleng2005Logo.jpg>

[29] DARPA. *Urban Challenge* [online]. 21.04.2008 [cit.17.05.2013]. Dostupné z:<http://archive.darpa.mil/grandchallenge/index.asp>

[30] Carnegie Mellon Tartan Racing. *Tartan Racing* [online]. 2007 [cit.17.05.2013]. Dostupné z:<http://www.tartanracing.org/>

[31] Carnegie Mellon Tartan Racing. *„Boss"* [online]. [cit.17.05.2013].

Dostupné z:<http://www.tartanracing.org/images/gallery/race5.jpg>

[32] URMSON, Chris. The self-driving car logs more miles on new wheels. *Google Official Blog* [online]. 07.08.2012 [cit.17.05.2013]. Dostupné z:<http://googleblog.blogspot.hu/2012/08/the-self-driving-car-logs-more-miles-on.html>

[33] ACKERMAN, Evan. Google's Autonomous Car Takes To The Street. *IEEE Spectrum* [online]. 13.10.2010 [cit.17.05.2013]. Dostupné z:<http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/googles-autonomous-car-takes-to-the-streets>

[34] GUIZZO, Erico. HowGoogle's Self-Driving Car Works. *IEEE Spectrum* [online]. 18.10.2011 [cit.17.05.2013]. Dostupné z:<http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works>

[35] IVO, Jordán. *HIMOTO MONSTER EMXT-1* [online]. [cit.18.05.2013]. Dostupné z:<http://www.himoto.cz/zbozi/5414/HIMOTO-MONSTER-EMXT-1-4x4-RTR-1-10--2-4GHz-.htm>

[36] beagleboard.org. *BeagleBoard-xM Rev C System Reference Manual* [HTML document]. 04.04.2010 [cit.19.05.2013]. Dostupné z:<http://beagleboard.org/static/BBxMSRM_latest.pdf>

[37] beagleboard.org. *BeagleBoard-xM diagram* [online]. 01.06.2011 [cit.19.05.2013]. Dostupné z:<http://farm5.staticflickr.com/4068/4644432552_4d3d5d4d29_o.png>

[38] beagleboard.org. *BeagleBoard-xM Product Details* [online]. 01.06.2011 [cit.19.05.2013]. Dostupné z:<http://beagleboard.org/hardware-xM>

[39] Canonical Ltd.. *Instalace* [online]. 16.08.2012 [cit.19.05.2013]. Dostupné z:<http://wiki.ubuntu.cz/instalace>

[40] eLinux.org. *BeagleBoardUbuntu* [online]. 13.05.2013 [cit.19.05.2013]. Dostupné z:<http://elinux.org/BeagleBoardUbuntu>

[41] ZAMAN, Tim. *[Linux] BeagleBoard xM Installation* [online]. 17.05.2011 [cit.19.05.2013]. Dostupné z:<http://www.timzaman.com/?p=1114>

[42] Emgu CV. *Main Page* [online]. 16.09.2012 [cit.20.05.2013]. Dostupné z:<http://www.emgu.com/wiki/index.php/Main_Page>

[43] Emgu CV. *Shape (Triangle, Rectangle, Circle, Line) Detection in CSharp* [online]. 05.01.2012 [cit.10.05.2013]. Dostupné z:<http://www.emgu.com/wiki/index.php/Shape_%28Triangle,_Rectangle,_Circle,_Line%29_Detection_in_CSharp>

[44] opencv dev team. *Hough Circle Transform* [online]. 05.04.2013 [cit.10.05.2013]. Dostupné z:<http://docs.opencv.org/doc/tutorials/imgproc/imgtrans/hough_circle/hough_circle.html>