VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

MOBILNÍ ROBOT ŘÍZENÝ KINECTEM

SEMESTRÁLNÍ PRÁCE
SEMESTRAL THESIS

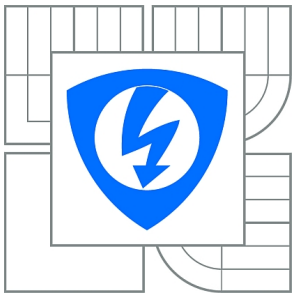
AUTOR PRÁCE
AUTHOR

Bc. MIROSLAV MÁLEK

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF RADIO ELECTRONICS

MOBILNÍ ROBOT ŘÍZENÝ KINECTEM

KINECT FOR CONTROLLING OF MOBILE ROBOT

SEMESTRÁLNÍ PRÁCE

SEMESTRAL THESIS

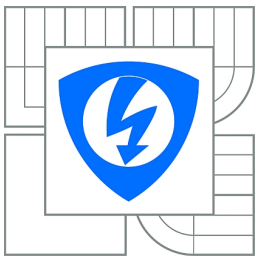
AUTOR PRÁCE — Bc. MIROSLAV MÁLEK

AUTHOR

VEDOUCÍ PRÁCE — Ing. ZBYNĚK FEDRA, Ph.D.

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student:	Bc. Miroslav Málek	***ID:***	115223
Ročník:	2	***Akademický rok:***	2012/2013

NÁZEV TÉMATU:

Mobilní robot řízený KINECTem

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s rozhraním a možnostmi zařízení MS KINECT. Prostudujte potřebné informace a případné ovladače na platformě ARM. Otestujte propojení a řízení KINECTu z vlastní aplikace. Zhodnoťte potřebný HW pro využití periferie a nároky na její řízení.
Sestavte a otestujte možnosti řízení podvozku robota na základě údajů získaných z KINECTu. Pokuste se vytvořit jednoduchou aplikaci na detekci překážek a zhodnoťte případné reakce na překážku.
Vytvořte kompletní systém mobilního robota s řízením využívajícím výsledky předchozí práce.
Zhodnoťte možnosti takovéto platformy a případné vhodné doplnění senzorické výbavy robota.

DOPORUČENÁ LITERATURA:

[1] HAGEN, W.The Definitive Guide to GCC. 2nd ed. New York: Apress, 2006.

[2] -, DIY Kinect Hacking [online]. ladyada.net, 2010 - [cit. 16.12.2010]. Dostupné na: http://ladyada.net/learn/diykinect/

Termín zadání:	11.2.2013	***Termín odevzdání:***	24.5.2013

Vedoucí práce: Ing. Zbyněk Fedra, Ph.D.
Konzultanti diplomové práce:

prof. Dr. Ing. Zbyněk Raida
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Tato práce se zabývá návrhem mobilního robota, který je řízen pohybovým zařízením MS Kinect. Pohyb robota je řízen hloubkovými daty, které jsou zpracovány vhodným ARM procesorem. Dále je v práci navržen modul pro sériovou komunikaci mezi procesorem a podvozkem robota. Pro uživatelský počítač a ARM procesor jsou navrženy softwarové aplikace, které umožňují pracovat s jednotlivými částmi robota a testovat jejich možnosti. V poslední části je v práci uvedena finální podoba robota ovládaného softwarem ARM procesoru. Robot se dokáže pohybovat mezi překážkami bez toho aniž by do některé narazil.

KLÍČOVÁ SLOVA

MS Kinect, BeagleBoard-xM, robot, libfreenect, OpenNI.

ABSTRACT

This project deals with design of a mobile robot controlled by MS Kinect. The movement of the robot is driven by depth data which is processed with a suitable ARM processor. There is a module designed for serial communication between the processor and the robot chassis. For user computer and ARM processor there are developed software applications to control each part of the robot as well. Finally, this project contains form of the built robot controlled by a ARM processor software. The robot has the ability of controlled movement between obstacles. This allows the robot to not come into contact with any obstacle.

KEYWORDS

MS Kinect, BeagleBoard-xM, robot, libfreenect, OpenNI.

MÁLEK, M. *Mobilní robot řízený MS KINECTem*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2013. 79 s. Vedoucí práce byl Ing. Zdeněk Fedra, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Mobilní robot řízený MS KINECTem“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

Poděkování

Jako autor diplomové práce bych rád poděkoval vedoucímu této práce Ing. Fedrovi, Ph.D. za velmi užitečnou metodickou pomoc, cenné rady při zpracování diplomové práce a za zapůjčení BeagleBoardu-xM Rev.C. Dále bych chtěl poděkovat doc. Ing. Frýzovi, Ph.D. za vypůjčení čtyřkolového podvozku. Drew Fisherovi za cenné informace při práci s Kinectem, uživateli IRC s přezdívkou AV500, za sdílení zkoušeností s minipočítačem BeagleBoard-xM a všem, kteří mě v mé práci podporovali.

V Brně dne ....

(podpis autora)

Výzkum realizovaný v rámci této diplomové práce byl finančně podpořen projektem
CZ.1.07/2.3.00/20.0007 **Wireless Communication Teams**
operačního programu **Vzdělávání pro konkurenceschopnost**.

Finanční podpora byla poskytnuta Evropským sociálním fondem
a státním rozpočtem České republiky.

Tento příspěvek vzniknul za podpory projektu CZ.1.07/2.3.00/20.0007 WICOMT,
financovaného z operačního programu Vzdělávání pro konkurenceschopnost

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Při konstrukci robotů, ovládaných MS Kinectem, se běžně používá způsob, který je založen na principu, kdy je MS Kinect připojen k uživatelskému počítači, který zpracovává a vyhodnocuje vstupní data. Tato vyhodnocená data se poté bezdrátovou technologií (Wifi, Bluetooth...) přenášejí do robota, jehož součástí je procesor, který na základě přijatých dat ovládá kola, pásy, nebo jiné jeho části. Tento způsob ovládání znemožňuje robotu samostatně zpracovávat vstupní data - vše je řízeno člověkem.

Mým úkolem bude otestovat vlastnosti MS Kinectu. Navrhnout řešení pro ovládání podvozku a sestavit robota s tak výkonným ARM procesorem, který by umožňoval samostatně zpracovávat vstupní data z Kinectu bez jakékoliv lidské pomoci.

Diplomová práce je členěna do sedmi kapitol. První kapitola podává informace o zařízení MS Kinect, jeho historii, použité technologii a jeho vlastnostech. Druhá kapitola popisuje software, pomocí kterého je možné získávat data z Kinectu. Následující kapitola je věnována popisu a konstrukci podvozku robota, ve které nechybí návrh Sériového modulu a příslušný testovací program. Pro získání představy o možnostech Kinectu je ve čtvrté kapitole vytvořen a popsán testovací program, který s pomocí běžného počítače komunikuje s Kinectem. Program vykresluje hloubková data (Depth data) a RGB video data. V následující části je práce soustředěna na popis použitého minipočítače, který tvoří jádro celého robota. V předposlední kapitole je uveden vytvořený robot. Tato část důkladně popisuje vytvoření Sériového modulu pro komunikaci s podvozkem, instalaci a vytvoření knihovny pro obvody FTDI a chování robota. Na závěr jsou shrnuty vlastnosti robota, hodnocení a možné úpravy pro dosažení lepších výsledků.

1 MS KINECT

Kinect je pohybový senzor (Motion Sensor), vyvinutý firmou Microsoft. Poprvé byl Kinect představen na Entertainment Software Association (ESA) konferenci, pod označením Project Natal [1]. Oficiální vydání Kinectu proběhlo 11.4. 2010 v Severní Americe. Tento přístroj představuje cenově dostupné zařízení, umožňující záznam zvuku, obrazu, vyhodnocení vzdálenosti a následné vyhodnocení získaných dat.

Obr. 1.1: Představení projektu Natal 6.1. 2009 (převzato z [1]).

1.1 Použitá technologie

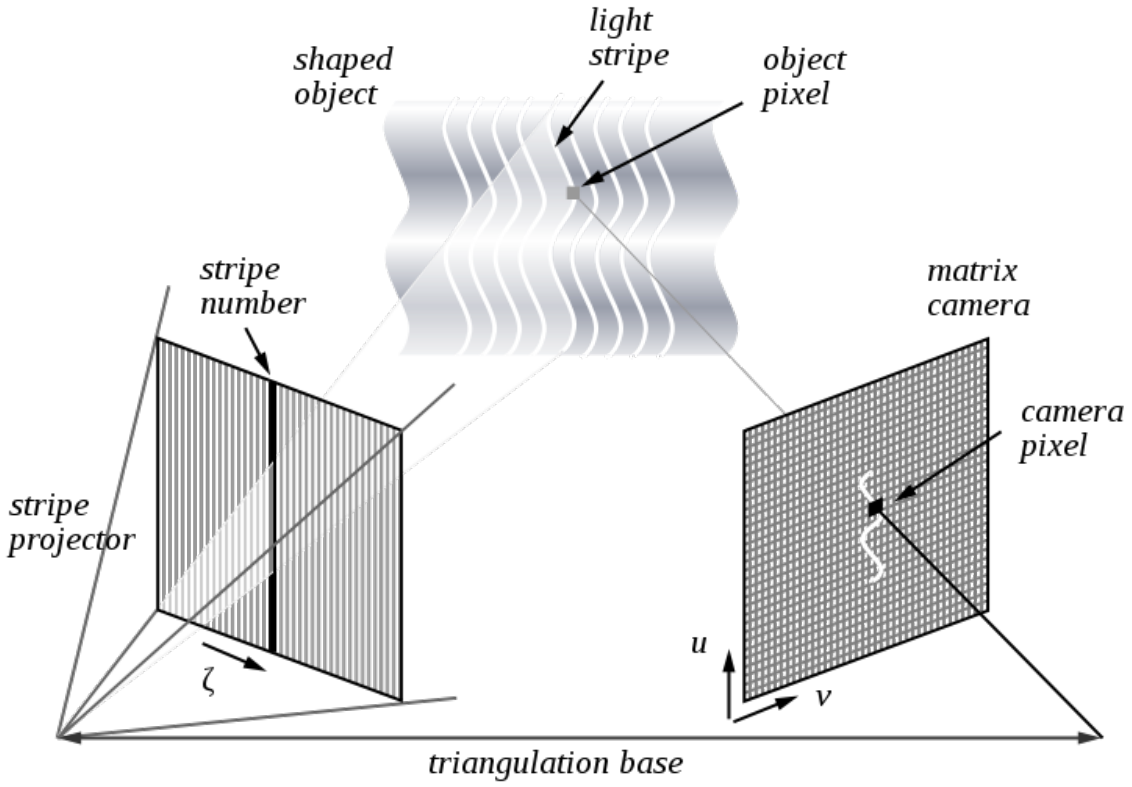
Existuje několik základních principů pro zpracovávání vzdáleností objektů. Techniky zpracování vzdáleností využívají vlastnosti různých druhů vlnění:

- Mikrovlnné vlny - civilní,vojenské radary,
- ultrazvukové vlny - např. lodní sonar pro mapování mořského dna využívá ultrazvukové vlny,
- světelné vlnění - technologie využívá světelného záření (v oblasti infra záření).

Protože Kinect využívá pouze světelného vlnění k určení vzdálenosti objektů, nemá smysl se zde dále zabývat mikrovlnným a ultrazvukovým vlněním.

1.1.1 Light Coding firmy Prime Sense

Jak již bylo zmíněno, Kinect je zařízení založené na principu měření vzdálenosti světelného záření. Podrobnosti této technologie nejsou věřejně přístupné [34]. Kinect nepoužívá technologii zvanou TOF (Time Of Fly - doba letu), ale technologii zvanou světelné kódování (Light Coding[36]). Tato technologie, podle poznatků Johna MacCornicka[34], používá principů Strukturovaného světelného scanování (Structured Light Scanning) v kombinaci s analýzou zaostření (Depth from focus) a principem Triangulace (Depth from stereo uses parallax). 3D skenování, založené na principu triangulace, lze snadno pochopit z obrázku 1.2. Obě dvě zařízení (kamera a zářič) jsou umístěny v jedné ose a vzdáleny od sebe, v případě Kinectu, přibližně 7,5 cm.

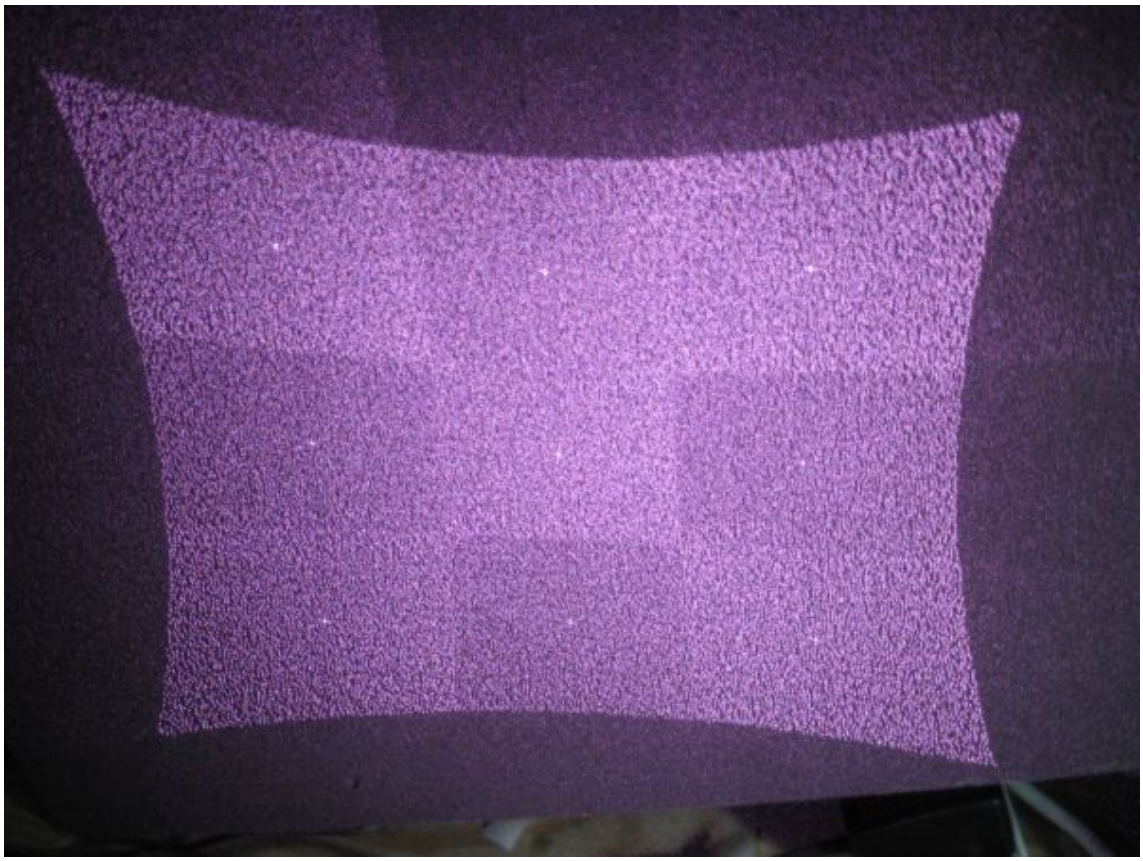


Obr. 1.2: Triangulační princip při 3D scanování (převzato z [13]).

Kamera disponuje optikou, s bodovou maskou zatímco zářič využívá masku tvořenou úzkými proužky. Při zapnutí zářiče dojde k proužkovému osvětlení plo-

chy před zařízením, které se při pohledu přes bodovou masku pro kameru jeví jako body. Tyto body jsou od sebe v osách x a y vzdálené o určitou délku, která závisí na vzdálenosti od zařízení.

Při nahlédnutí do patentových listů firmy Prime Sense Ltd. dostupných na [33] je možné vyčíst základní vlastnosti zařízení. Na objekt se promítá nekorelovaný vzor bodů (Uncorrelated pattern)[33], který je generovány infračerveným (Infrared - zkráceně IR) zářičem viz. obrázek č. 1.3. Na základně principu triangulace, IR senzor sejme snímek, který dále zpracovává.

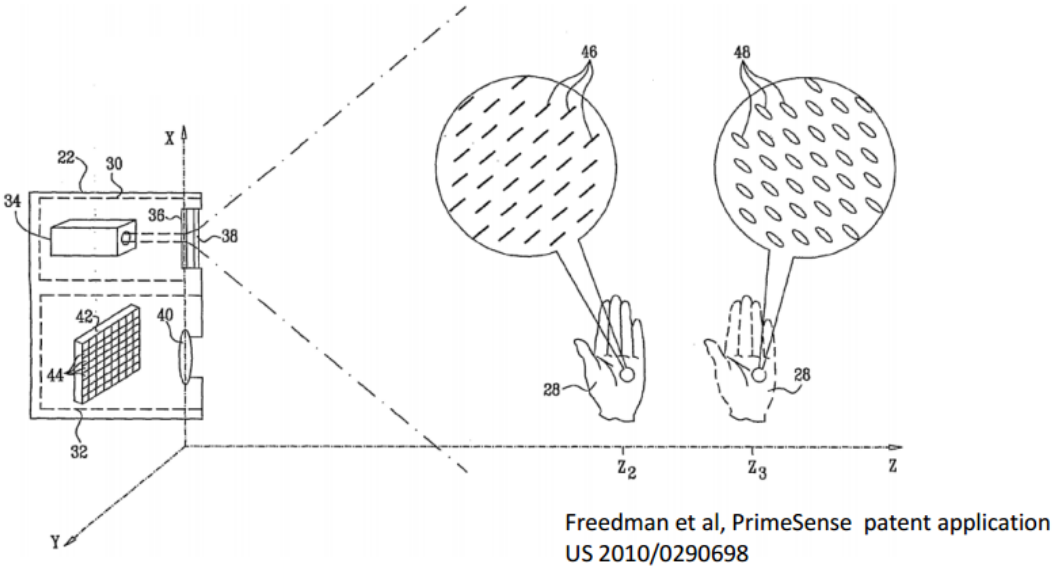


Obr. 1.3: Nekorelovaný obrazec vytvořený Kinectem (převzato z [37]).

3D mapa objektu se vytváří způsobem, kdy se ozářená plocha rozdělí na několik částí (v případě Kinectu na 9 částí), které jsou jasně viditelné na obrázku výše. Tyto části se následně porovnávají s referenčním snímkem, který odpovídá známé vzdálenosti, buď korelací (Image Correlation), nebo jinou známou metodou pro porovnávání obrázků. Vzdálenost je ve výsledku určena z posunu sejmutého snímku vůči referenčnímu [33].

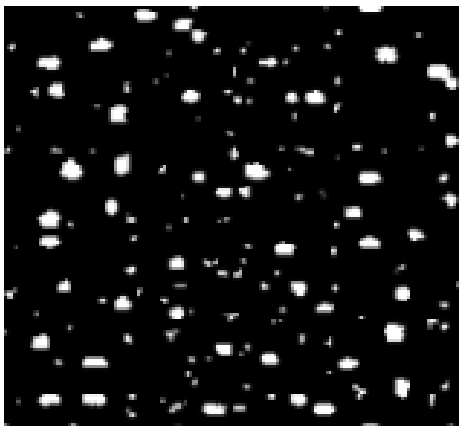
Podle pana MacCornicka, technika Depth from focus byla vymyšlena tak, že přístroj dosahuje daleko větší přesnosti, oproti jiným zařízením, které využívají tuto techniku [34]. Body, emitované IR zářičem, jsou s rostoucí vzdáleností od přístroje více

rozmazány. Zároveň se před IR zářičem nachází vhodně umístěné astigmatické čočky s rozdílnou ohniskovou vzdáleností v x-ovém a y-ovém směru. Promítnutý bod se po průchodu těmito čočkami stává elipsou, jejíž orientace závisí na vzdálenosti viz. obrázek č.1.4.



Obr. 1.4: Použití astigmatických čoček (převzato z [34]).

Na následujícím obrázku 1.5 je patrná deformace vyzářených bodů zářiče.



Obr. 1.5: Ozářený prostor IR zářičem Kinectu (převzato z [34]).

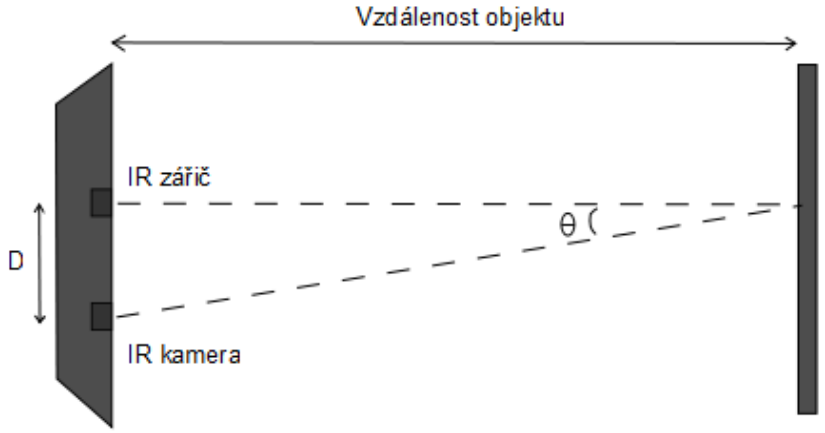
1.2 Hardware MS Kinectu

MS Kinect využívá výše zmíněnou technologii a zpracovává vzdálenosti objektů od Kinectu jako tzv. hloubková data (Depth Data). K tomuto účelu mu slouží speciální infračervený zářič a kamera.

1.2.1 Hloubkový senzor

Hloubkový senzor (Depth sensor) se skládá z infračerveného zářiče (IR emittor) a CMOS senzoru infračerveného záření (IR kamera) s rozlišením 640 x 480 bodů. IR zářič představuje laserovou diodu o vlnové délce 830nm jejíž výstupní úroveň výkonu je konstantní [38]. Zářič a senzor jsou umístěny na čelní straně Kinectu ve vzdálenosti přibližně 7,5 cm od sebe. V okamžiku spuštění hloubkového senzoru se IR zářič zapne a ozáří prostor před Kinectem. Zároveň dojde k zapnutí infračerveného snímače, který ozářený prostor snímá. Informace získané senzorem vyhodnocuje čip PS1080, který z dat vytvoří hloubková data, která obsahují informace o vzdálenosti příslušných bodů. PS1080 dokáže data snímat rychlostí 30 snímků/sekund s úhlem pohledu (Field Of View) hloubkového senzoru 58° ve horizontální rovině a 45° ve vertikální. Hloubková data jsou vytvořena s 11-ti bitovou přesností, což představuje 2048 různých hodnot hloubkových dat. Snímání hloubkového senzoru není lineární, ale logaritmické - objekty vzdálenější od Kinectu jsou zpracovány s nižší přesností, než objekty v těsné blízkosti Kinectu.

Mikkel Viager[30] změřil vlastnosti Kinectu při vyhodnocování vzdálenosti a odvodil vztahy 1.1 až 1.4. Svým měřením dokázal, že mezi 11 bitovou hodnotou N, kterou vrací Kinect, a úhlem Θ platí vztah 1.2:



Obr. 1.6: Princip měření vzdálenosti Kinectem

$$\Theta = arctg \cdot \left(\frac{D}{L}\right) \qquad [rad] \tag{1.1}$$

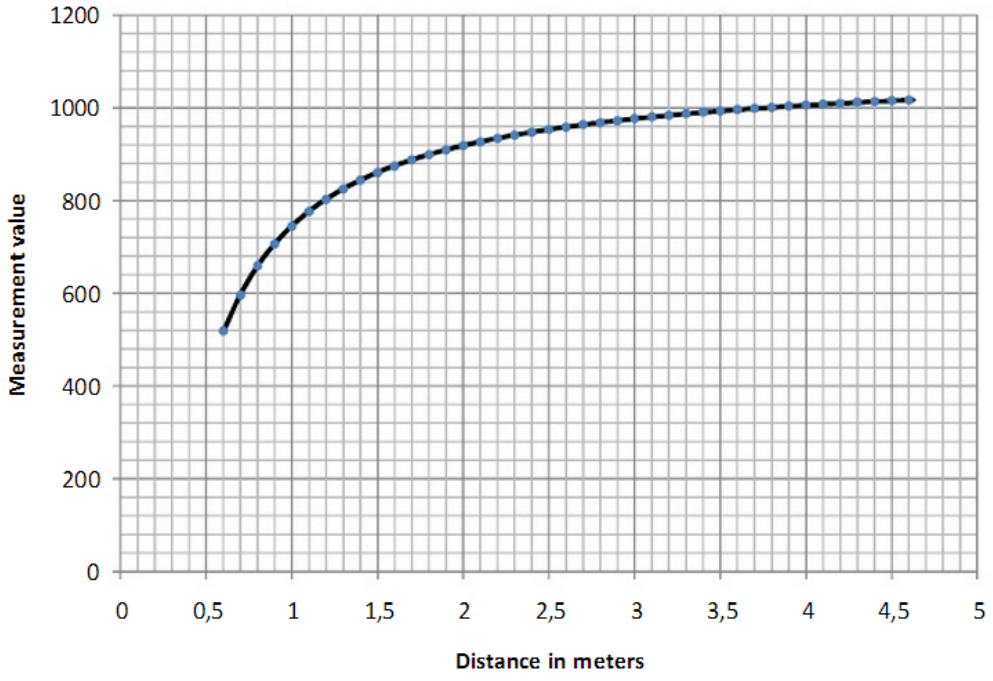
Kde:
D je vzdálenost IR zářiče a IR kamery Kinectu. Tedy platí: $D = 0,075m$. Parametr L udává vzdálenost Kinectu od měřeného objektu a Θ je paralaxe mezi kamerou a IR zářičem.

$$N = -4636,3 \cdot \Theta + 1092,5 \qquad [-] \tag{1.2}$$

Po dosazení vztahu 1.1 do vztahu 1.2 a následnou úpravou vznikne vztah 1.3 pro výpočet vzdálenosti Kinectu od objektu v závislosti na hodnotě N. Průběh této závislosti je uveden na obrázku 1.7, který se nalézá na další straně.

$$N = -4636,3 \cdot arctg\left(\frac{0.075}{L}\right) + 1092,5 \qquad [-] \tag{1.3}$$

$$L = -\frac{0,075}{tg(0,0002157 \cdot N - 0,2356)} \qquad [m] \tag{1.4}$$



Obr. 1.7: Závislost měřené hodnoty N na vzdálenosti objektu L (převzato z [30]).

1.2.2 RGB kamera

RGB kamera představuje 8-bitový VGA snímač s Bayerovým filtrem[38], která dosahuje maximálního rozlišení 640 x 480 pixelů při rychlosti snímání 30-ti snímků za sekundu. Kamera je umístěna mezi infračerveným zářičem a snímačem. Úkolem této kamery je snímat prostředí před Kinectem ve skutečném zobrazení, což se dá použít jako doprovodná video sekvence při hraní her na herní konzoli Xbox, nebo jako webová kamera k počítači.

1.2.3 Motorizovaná základna

Motorizovaná základna Kinectu je tvořena malým bipolárním servomotorem, který umožňuje natáčet zařízení v ose x přibližně +/-27 stupňů.

1.2.4 Zvukové snímače

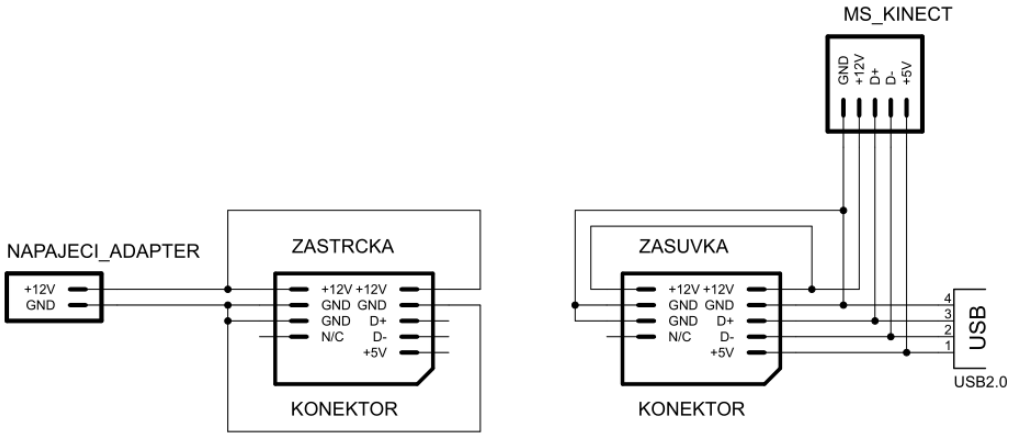
Přístroj je doplněn polem čtyř mikrofonů s 16-bitovými ADC (Analog to Digital Converter) převodníky, odolnými vůči akustickému echu a nežádoucímu šumu. Zaznamenávaný audio formát je modulovaný 24-bitovou pulsně šířkovou modulací (Pulse Width Modulation), která je vzorkovaná generátorem o kmitočtu 16 kHz.

Tři mikrofony jsou umístěny ve spodní části na pravé straně a jeden na levé. Tato koncepce umožňuje snadnější rozpoznávání řeči, určení v prostoru a potlačení případného rušivého šumu.

1.2.5 Konektor a napájení

Původně byl Kinect navržen pro připojení k herní konzoli Xbox360 přes USB rozhraní. USB sběrnice herní konzole však nedokázala Kinectu poskytnout dostatečný příkon, a proto byl vybaven speciálním konektorem s napájecím adaptérem viz. princip zapojení uvedený na obrázku 1.9. Konektor je vyšší, připomíná robustnější USB konektor, nebo Firewire, a jedna z jeho stran má zkosenou hranu. Konektor obsahuje několik pinů - D+,D-,+5V, GNG, které jsou shodné s běžným USB. Z důvodu daleko vyššího příkonu Kinectu, než dokáže USB sběrnice poskytnout, jsou v konektoru ještě piny pro připojení 12-voltového napětí, které je získáno ze síťového adaptéru. Adaptér se skládá z impulsního stejnosměrného zdroje o velikosti napětí 230V/12V a redukce z Kinect adaptéru na běžné USB2.0. Schéma zapojení konektoru ke Kinectu je patrné z následujícího schématu, které zobrazeno na obrázku 1.8.

Novější konzole Xbox360S již umožňují přímé připojení Kinectu a napájení obstarává konzole sama.

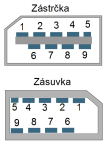


Obr. 1.8: Schéma zapojení konektoru.

Obr. 1.9: Zapojení Kinectu k herní konzoli Xbox360 [22].

Tab. 1.1: Popis a význam pinů konektoru.

Č. pinu	**Význam**	**Barva**	**Význam**
1	VCC	žlutá	+12V
2	D-	bílá	Data -
3	D+	zelená	Data +
4	GND	černá	0V
5	+5V	červená	+5V
6	VCC	žlutá	+12V
7	GND	černá	0V
8	GND	černá	0V
9	N/C	-	N/C



Obr. 1.10: Zapojení pinů Kinect konektoru - zástrčka a zásuvka.

1.3 Konkurenční zařízení

Pohybové ovládání není již záležitostí pouze herních konzolí. Objevují se také řešení pro běžné počítače.

Obr. 1.11: Prime Sense device, Kinect a Asus (převzato z [19]).

1.3.1 Microsoft Kinect pro Windows

Zatímco prvotní verze Kinectu byla určena pro herní konzoli Xbox, novější Kinect tzv. Microsoft Kinect for Windows, je určen především pro počítačové vývojáře. Microsoft jej uvedl do prodeje spíše v tichosti, avšak jeho potenciál je mnohem větší [16]. Tento typ Kinectu podporuje tzv. Near mode[3], což je mód umožňující snímat objekty z minimální vzdálenosti 40 centimetrů do 4 metrů.

1.3.2 Asus Xtion

Jedním z konkurenčních zařízení Kinectu je pohybové zařízení od firmy Asus - senzor Xtion. Xtion se k počítači, nebo notebooku připojuje přes rozhraní USB 2.0. Součástí dodávky Xtion je i speciální software, který obsahuje multimediální, sociální a zábavní aplikace. Tento pohybový senzor je určený pro notebooky nebo počítačové sestavy. Dokáže snímat lidskou postavu od 0,8 metrů do 3,5 metrů od senzoru[39]. Efektivní úhel snímaný v horizontální a vertikální rovině je stejný jako v případě Kinectu - 58° horizontální a 45° vertikální.

Tab. 1.2: Parametry senzoru Xtion firmy ASUS [15]

Dosah	0,8 m až 3,5 m
Rozhraní	USB 2.0
Zorné pole	58° H, 45° V, 70° D
Software	Xtion Portal
Rozměry	18 cm x 3,5 cm x 5 cm

1.3.3 Pohybové zařízení firmy Prime Sense Ltd.

Jedná se o pohybové zařízení vyvinuté firmou Prime Sense[1], jehož jádro tvoří čip PS1080. Je to stejný čip jako v případě MS Kinectu. Zařízení má obdobné vlastnosti, jen design je řešený jiným způsobem, což je zřejmé z následujícího obrázku 1.12.

Obr. 1.12: "Kinect"od firmy Prime Sense (převzato z [23]).

1.3.4 I-dong

V listopadu roku 2010 bylo na čínském špičkovém veletrhu Hi-Tech Fair (CHTF) představeno zařízení I-dong, které vytvořila čínská firma Taishan Online Technology Co Ltd [24]. Čínský I-dong je svým vzhledem velice podobný MS Kinectu, avšak nabízí herním nadšencům nové možnosti ovládání her, počítačů a set-top boxů.

I-dong by se dalo chápat jako spojení technologie Kinectu a PS Move[2]. Tvoří jej malá černá krabička s IR kamerou, IR zářiči a speciální ruční ovladač. Celá tato koncepce může fungovat zvlášť, nebo dohromady. Lze například v rukou držet

[1]Izraelská firma založená v roce 2005 a v roce 2011 obdržela ocenění jako jedna z 50 nejlepších světových inovativních společností [4].

[2]Herní pohybové zařízení využívající speciálních ovladačů pro získání potřebných dat vyvinuté firmou Sony Computer Entertainment (SCE).

ovladač a před senzorem se pohybovat tím hrát tenisovou hru, kterou hráč uvidí před sebou na obrazovce. Cena I-dongu se na čínském trhu pohybuje okolo 222.89$ [24].

Obr. 1.13: Kinect čínské výroby - I-dong [24].

Obr. 1.14: Součást I-dongu - ruční ovladač [25].

1.3.5 Leap Motion

Leap je malá krabička, kterou je potřeba připojit k USB počítače. Poté už stačí jen pohybovat prsty rukou a systém vše zachytí a převede do virtuálního 3D prostoru. Leap tedy dokáže spočítat nejen drobné pohyby jednotlivých prstů, ale i hloubku pohybu. Zároveň dokáže sledovat i neživé objekty - třeba tužku.

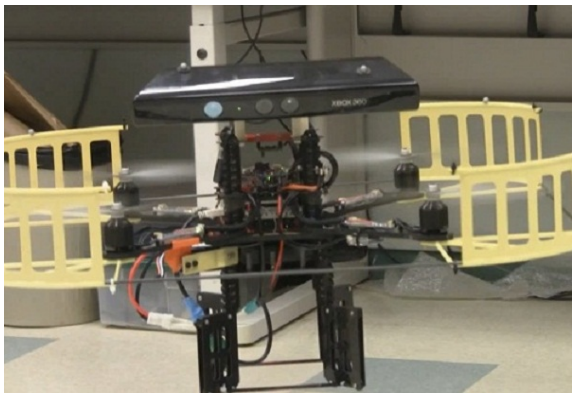
1.4 Využití zařízení

Kinect našel široké využití nejen v herním průmyslu, ale také v lékařství, telekomunikaci, počítačové grafice a robotice.

1.4.1 Čtyř-rotorová helikoptéra Patricka Bouffarda

Patrick Bouffard použil Kinect a model čtyř-rotorové helikoptéry (také nazývané quadrotor) k pohybu helikoptéry po prostoru. Helikoptéra je schopna autonomního

pohybu podél předem definovaných bodů[27]. V případě detekce překážky se quadrotor zastaví a počká než se překážka z cesty odstraní. Aktuální výška helikoptéry a pozice překážek je získávána na základě dat poskytovaných Kinectem [26].



Obr. 1.15: Quadrotor a Kinect v akci [28].

1.4.2 Pohybové zařízení a použití v lékařství

Původně zamýšlené využití Kinectu se z oblasti her rozšířilo i do oboru lékařství [27]. Při lékařských zákrocích je určitě důležitá sterilita chirurgových rukavic, aby se předešlo možným komplikacím. Vzhledem k tomu, že chirurg pracuje s velkým množstvím nástrojů, je zachování sterility jeho rukavic náročné.

Existuje projekt zvaný Virtopsy (Bernské lékařské univerzity ve Švýcarsku), jenž se zabývá vytvořením softwaru, který za pomoci Kinectu, umožňuje při operacích využít přídavné specifické funkce, vedoucí ke zvýšení kvality operací. Například pomocí definovaných gest a povelů může chirurg ovládat počítač s pacientovými daty, prohlížet rentgenové snímky a další.

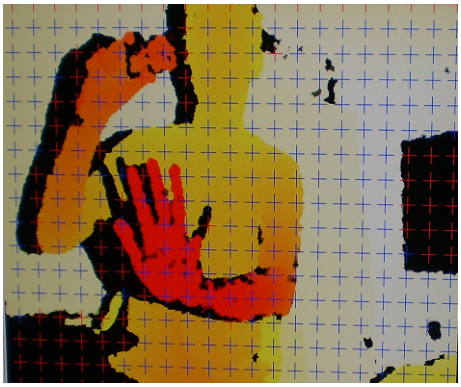
1.5 Nevýhody zařízení

Tato část je věnována nedostatkům Kinectu, které jsem během práce s přístrojem vypozoroval. MS Kinect se může zdát dokonalým cenově dostupným zařízením, má však své skryté nedostatky, které znemožňují použití například na venkovních prostranstvích.

1.5.1 Nežádoucí "stín"ve zpracovaném obraze

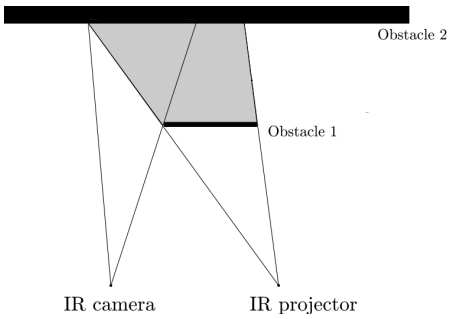
Na obrázku 1.16 je vidět jeden snímek zpracovaný z Kinectu. Při jeho bližším zkoumání si nejde nevšimnout vzniklého černého "stínu", který je tím větší, čím

blíž je snímaný objekt u Kinectu. Pokud je objekt vzdálený více jak jeden metr od zařízení, nelze na snímku prakticky pozorovat žádný "stín". Prostě a jednoduše splyne s objektem.



Obr. 1.16: Snímek hloubkové mapy získané z Kinectu.

Podle autorů OpenKinect[7] je příčinou vzniku těchto "stínů"princip snímací technologie Kinectu. Černá místa, která vypadají jako "stíny"originálních objektů, jsou místa bez hloubkových dat. V kapitole "Použitá technologie"byl objasněn princip snímání hloubkových dat. Avšak za určitých okolností tento princip snímání vykazuje zásadní nedostatek viz. obrázek 1.17.



Obr. 1.17: Způsob vzniku stínů ve snímku (převzato z [7]).

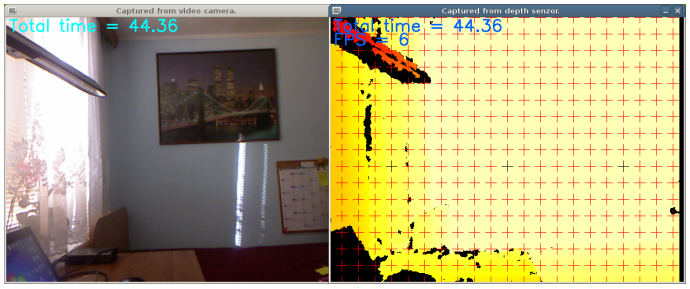
Jak zářič, tak kamera mají svoje vlastní pole pohledu (Field Of View). V některých místech se však tyto pole neprolínají a tím vznikají chyby při zpracování.

Kamera snímá prostor ozářený zářičem, viz obr.1.17. Pokud dojde k tomu, že infračervený zářič neosvítí celou plochu objektu, kterou kamera snímá, vzniknou tím místa bez hloubkových dat. Kamera tyto místa snímá, ale nenalezne žádné odražené IR paprsky. Ve výsledném zpracování obrazu dojde k tomu, že čip Kinectu tuto oblast, bez hloubkových dat, vyhodnotí jako oblast velice vzdálenou, a proto

ji přiřadí maximální hodnotu dat. V tomto případě odpovídá maximální hodnotě hloubkových dat černá barva.

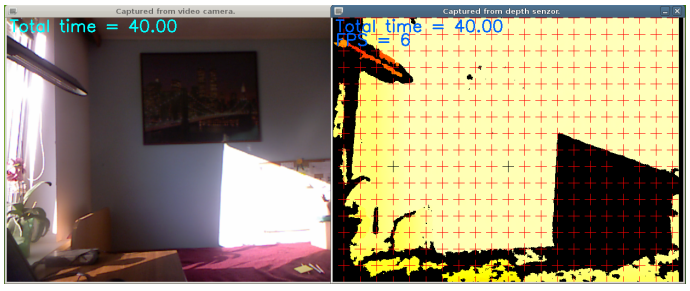
1.5.2 Vliv IR záření okolí

Jak již bylo uvedeno výše, Kinect snímá IR kamerou plochu, která je ozářená infračerveným zářičem a na základě získaných informací provádí korelaci s referenčním snímkem. Za ideálních podmínek je plocha ozářena pouze infračerveným zářičem bez jakýchkoliv dalších externích zdrojů IR. Ve skutečnosti na snímanou plochu dopadá záření z externích zdrojů. Jedním z takových zdrojů je například Slunce. Infračervené sluneční paprsky se při snímání Kinectem projeví jako nežádoucí rušení, které způsobuje nekorektnost při korelaci dat. Pro ukázku byly vytvořeny dva demonstrační snímky, na kterých je při prvním pohledu patrný rozdíl ve snímaných datech. Na obrázku 1.18 je vidět stěna za běžného pokojového osvětlení. V tomto případě probíhá vyhodnocování dat Kinectem spolehlivě a nedochází tak k nežádoucímu problémům způsobeným externím IR zářičem.



Obr. 1.18: Snímání prostředí při standardních podmínkách osvětlení prostoru.

Druhý obrázek 1.19 vykresluje ten samý prostor, jako v prvním případě, jen s tím rozdílem, že na část snímané plochy dopadá sluneční záření.



Obr. 1.19: Vliv externího IR záření na vyhodnocení hloupkových dat.

Tato část plochy je poté přístrojem mylně vyhodnocena a nabývá nesprávných hodnot ve výsledném hloubkovém snímku. V tomto případě nabývala ozářená část hodnot, které odpovídají vzdálenosti v intervalu 0-50 cm.

2 SOFTWARE PODPORUJÍCÍ PRÁCI S KINECTEM

2.1 Vývojový software pro práci s Kinectem

Existují čtyři hlavní knihovny pro ovládání Kinectu [6]:

- OpenKinect libfreenect - Open Source knihovna vytvořená nadšenci o problematiku Kinectu,
- OpenNI - framework vytvořený Prime Sense Ltd.,
- CLNUI - projekt zaměřený pouze pro platformu Windows, umožňuje práci více zařízení MS Kinect
- Microsoft Kinect for Windows SDK - oficiální softwarový vývojový kit od společnosti Microsoft

2.2 OpenKinect a knihovna libfreenect

Libfreenect je knihovna, která byla vytvořena způsobem reverzního inženýrství. Vývojáři se snažili objasnit komunikaci mezi Kinectem a herní konzolí Xbox. Své poznatky podrobně zdokumentovali[7] a nasdíleli ve své komunitě OpenKinect. Pro komunikaci s Kinectem využívá knihovna libfreenect knihovnu libusb, která dává aplikaci snadný přístup k USB zařízení na operačních systémech jako je Windows, nebo Linux [8]. Hlavní výhodou libfreenect je snadná implementace a možnost ovládat všechny části Kinectu (včetně motoru). Nevýhodou však je nemožnost použít speciálního softwaru, tzv. *Middleware NITE*, pro zpracování gest a pohybů uživatele.

Knihovna je široce zpracovaná, umožňuje developerům použít různé programovací jazyky: C, C++, C#, Java, Python a další. Libfreenect se skládá z několika částí, které lze použít při vývoji aplikací. Každá z těchto částí funguje samostatně a liší se svoji syntaxí a funkcí:

- Libfreenect asynchronní knihovna pro C/C++,
- libfreenect synchronní knihovna pro C/C++,
- libfreenect objektově orientovaná knihovna pro C++.

2.2.1 Objektově orientovaná knihovna libfreenect pro C++

Objektově orientovaná knihovna libfreenect pro použití společně s jazykem C++ je definována v hlavičkovém souboru "libfreenect.hpp". V tomto souboru se nachází definice třech tříd:

- Freenect::Noncopyable,
- Freenect::Freenect T ,
- Freenect::FreenectDevice.

Při vytváření softwaru s touto knihovnou se musí postupovat následovně [7]:

1. Vloží se hlavičkový soubor *libfreenect.hpp*,
2. musí se vytvořit třída, která bude potomkem třídy Freenect::FreenectDevice,
3. definují se patřičné call-back funkce pro získávání dat.

2.2.2 Libfreenect asynchronní knihovna pro C/C++

Asynchronní knihovna libfreenect umožňuje asynchronní přístup k částem Kinectu. Funkce je taková, že se nejprve includuje příslušný hlavičkový soubor *libfreenect.h*. Následně se přiřadí knihovně call-back funkce, které předávají data z Kinectu. Po přiřazení call-back funkcí se celý Kinect inicializuje a v nekonečné smyčce se volá *Update()* funkce Kinectu. V momentě, kdy má Kinect dostupná data (RGB video, depth data), je zavolána příslušná call-back funkce, se kterou se předají získaná data. Náplní vývojáře je, aby výsledná data vhodným způsobem zpracoval. Po ukončení práce s Kinectem je nutné zavolat funkci pro deinicializaci Kinectu, jinak hrozí vznik děr v paměti tzv. "Memory Leaks".

Ukázka použití asynchornní knihovny libfreenect:

```
#include <iostream>
#include <libfreenect.h>

freenect_device * device = NULL;
freenect_context * context = NULL;
void Depth_CallBack(freenect_device * device, void * depth, uint32_t
timestamp)
{
 /* Zde se libovolně zpracovávají Depth data z Kinectu */
}
```

```
int main (int argc, char **argv)
{ /* Inicializace a spuštění zařízení */
 if(freenect_init(&context,NULL)<0){ printf("Inicializace selhala");
  return 1;}
 if(freenect_open_device(context, &device,0)<0) {printf"Nelze spustit
  zařízení"; return 1;}
 /* Nastavení call back funkce pro ziskavani Depth dat */
 freenect_set_depth_callback(device, Depth_CallBack);
 freenect_start_depth(device);
 while(freenect_process_events(context)>=0);
 freenect_stop_depth(device);
 freenect_close_device(device);
 freenect_shutdown(context);
 return 0;
}
```

2.2.3 Libfreenect synchronní knihovna pro C/C++

Pro práci s touto knihovnou se do vyvíjeného projektu musí vložit *libfreenect-sync.h* namísto *libfreenect.h*. Běžící program volá vybrané funkce a tyto funkce vrací dostupná data. Výhodou synchronní knihovny je dosáhnutí větší rychlosti zpracovávaných dat na málo výkonných zařízeních. Rovněž její implementace je snazší, což je patrné z následující ukázky:

```
#include <stdlib>
#include <libfreenect-sync.h>

int main(int argc, char **argv)
{
 short * depth; /* Ukazatel na pole dat získaných
  z Kinectu (640*480*3) */
 uint32_t ts;
 freenect_sync_get_depth((void**)&depth,&ts,0,FREENECT_DEPTH_11BIT);
 freenect_sync_stop();
 return 0;
}
```

Následující výčet představuje nejdůležitější funkce synchronní knihovny libfreenect:

- freenect_sync_get_video - funkce pro získání dat z RGB kamery,
- freenect_sync_get_depth - umožňuje předání depth dat,
- freenect_sync_set_led - nastavuje barvu indikační diody,
- freenect_sync_stop - přeruší práci s Kinectem.

2.2.4 Zpracování dat získaných z depth kamery Kinectu

Určení vzdálenosti objektů od senzoru Kinectu je důvod, proč se zařízení používá. Knihovna libfreenect, ať už synchronní, nebo asynchronní v argumentech funkcí vrací ukazatel na pole, které představuje hodnoty získané z depth senzoru. Tyto data však neodpovídají reálným vzdálenostem. Proto je nutné tyto hodnoty přepočítat na skutečnou vzdálenost v metrech. Existuje několik definic pro výpočet. Liší se od sebe použitým formátem depth dat. Pro 11 bitový formát dat platí vztah 2.1 z [7].

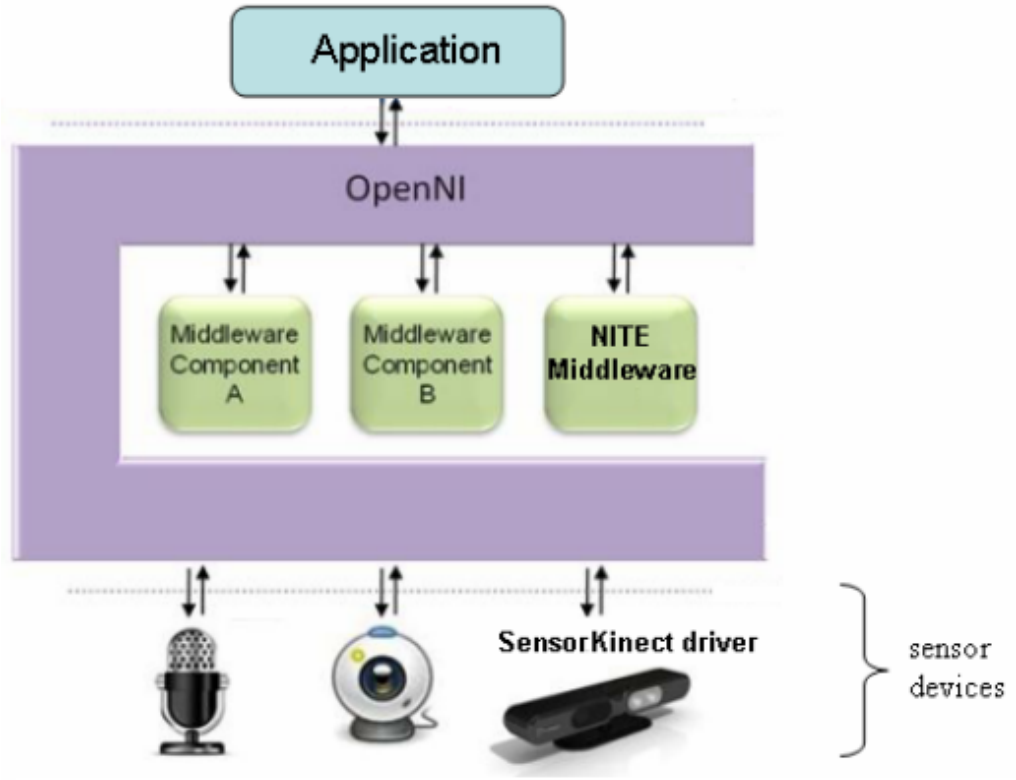
$$vzdalenost = 0,1236 \cdot tg\left(\frac{DATA}{2842,5} + 1,1863\right) \qquad [\mathrm{m}] \tag{2.1}$$

2.3 OpenNI - Open Natural Interaction

V září 2010 založila firma Prime Sense Ltd. neziskovou organizaci zvanou OpenNI[5], která se zaměřuje na práci s pohybovými zařízeními. K OpenNI se následně připojily a podporují vývoj firmy Willow Garagem Side-Kick, ASUS a AppSide.

OpenNI vydala volný OpenNI Framework, který přináší API pro práci s Kinectem a mezivrstvu pro zpracování pohybu *NITE*. Hlavními výhodami OpenNI jsou:

- Podpora operačních systémů Windows (Vista, XP), Linux a Mac OSX,
- plná podpora pro Unity 3D herní engine,
- podpora nahrávání/ přehrávání záznamů na/z pevného disku,
- podpora zařízení založených na čipu firmy Prime Sense a zařízení ASUS WAVI Xtion,
- *Middleware NITE* pro zpracování pohybů a gest.



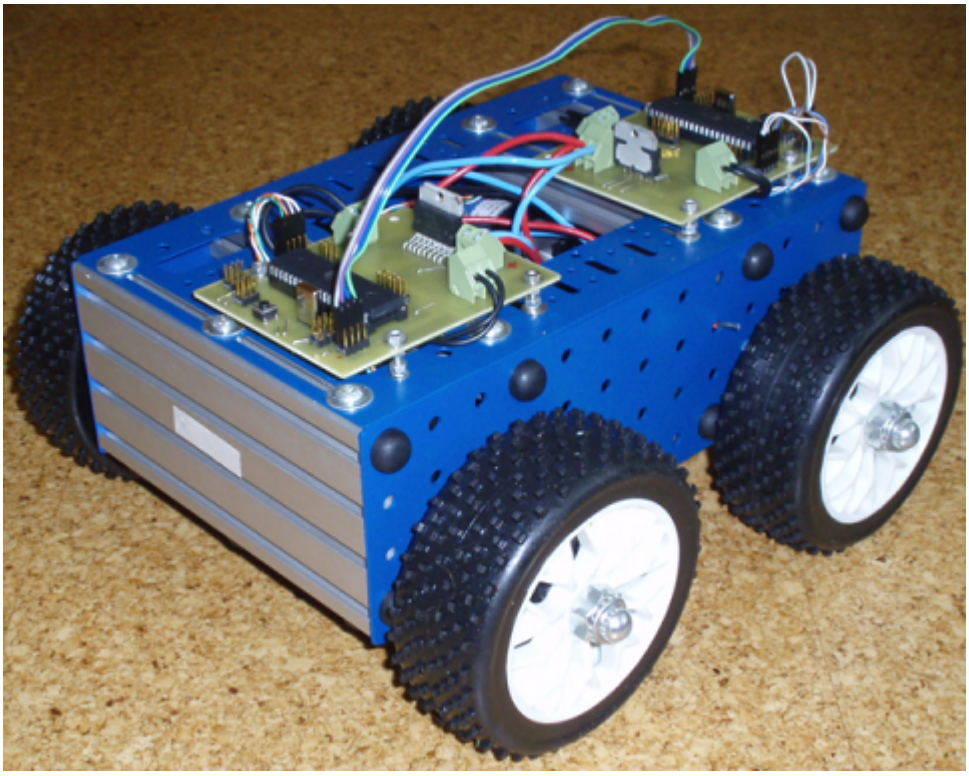
Obr. 2.1: Vrstvy OpenNI frameworku [6]

Struktura OpenNI Frameworku

OpenNI Framework (obr.2.1) je vícejazyčný, framework, který definuje API (Application Programming Interface) pro vývoj aplikací využívající přirozené iterakce lidského těla. OpenNI umožňuje přístup k různým částem zařízení prostřednictvím tvz. uzlů. Tyto uzly jsou ve frameworku definovány jako objekty různých zařízení, které zpracovávají rozdílná data.

3 PODVOZEK

Vybrat vhodný podvozek, který by svými vlastnostmi a technickými parametry nejlépe odpovídal požadavkům na budoucího mobilního robota, byl tvrdý oříšek. Na internetu je široká nabídka podvozků, lišící se výbavou, použitým materiálem, velikostí, ale hlavně cenou. Cena podvozků [1] začíná na ceně přibližně 1000 Kč za kus. Avšak tyto podvozky jsou vyráběné z plastových dílů [2] a jsou určeny pro malá zařízení. Robustnější podvozky vyrobené z kovových materiálů jsou na trhu dostupné přibližně od 3000 Kč a cena dále roste v závislosti na velikosti a nosnosti zařízení.



Obr. 3.1: Čtyřkolový podvozek [12].

3.1 Čtyřkolový podvozek s rozhraním I2C

Po pečlivém prostudování jsem použil podvozek na obr. č.3.1, označovaný jako MOB-3, zapůjčený od pana doc. Ing. Frýzy, Ph.D. Celý MOB-3 (kromě kol) je seskládán z nerezových a duralových profilů. Pohon obstarávají čtyři motory GM37, umístěné uvnitř celé konstrukce. Kola jsou k motorům připojena přes převodovku s převodovým poměrem $67,5:1$ označovanou jako GM37-82 [12].

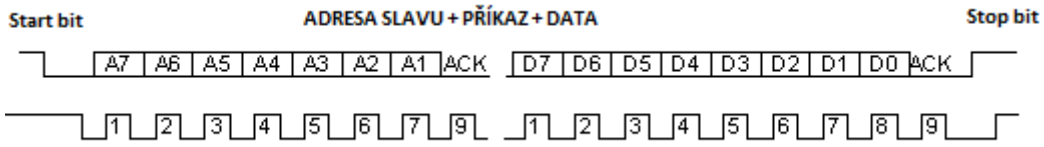
[1] Podvozek obsahuje pohonnou jednotku včetně převodovky.
[2] Nosnost nebývá často u takovýchto typů podvozků udávána.

Při použití této převodovky a 75% otáčkách dosahuje podvozek rychlosti $0,37m/s$ [35][12]. Na konstrukci podvozku jsou připevněna dvě zařízení typu slave. Zařízení typu master komunikuje se slavy přes sběrnici I2C, jejíž vlastnosti jsou popsány dále v této práci.

3.2 I2C protokol použitého podvozku

Aby bylo možné s řídícími mikroprocesory komunikovat po I2C sběrnici, je nutné znát její strukturu. Veškerá komunikace je řízena masterem, který představuje převodník USB/I2C. Nejprve zařízení typu master vyšle startovní bit, po kterém následují čtyři datové byty a vysílání ukončí stop bitem. Nutno podotknout, že mezi datovými byty se nachází ACK bity.

První z datových bytů má hodnotu adresy slave zařízení, pro které jsou data určena. Hodnota druhého bytu stanovuje povel a třetí a čtvrtý byte obsahují dodatečná data. Hodnoty povelů a možnosti všech čtyř bytů jsou uvedeny v tabulce 3.1.



Obr. 3.2: I2C protokol řízeného podvozku.

Tab. 3.1: Seznam možných povelů pro ovládání podvozku

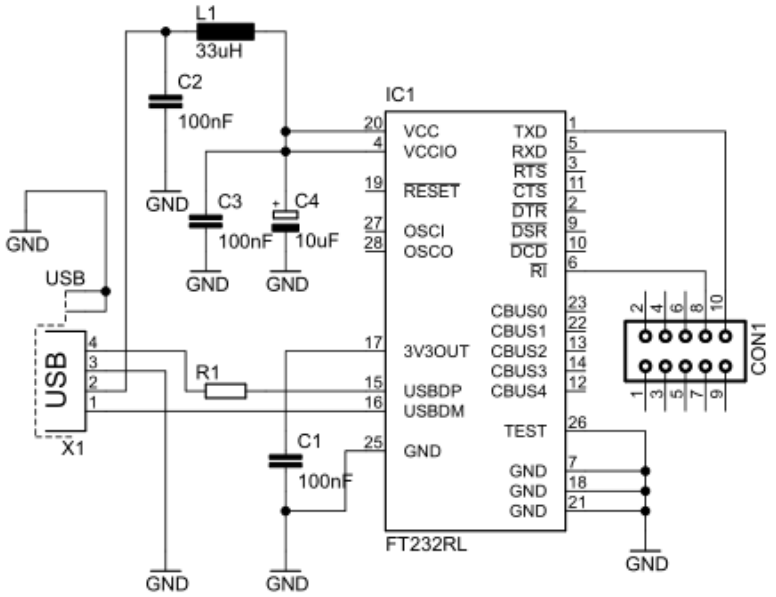
Povel	**1. byte**	**2.byte**	**3.byte**	**4.byte**
Směr vpřed	0x20/0x22	0x01	-	-
Směr vzad	0x20/0x22	0x02	-	-
Rychlost	0x20/0x22	0x04	-	0x00-0xFF
Zastavení	0x20/0x22	0x10	-	-

3.2.1 Testování podvozku

Pro testování zvoleného podvozku jsem využil integrovaný převodník USB-RS232 firmy FTDI - FT232RL. Jsou pro něj dostupné ovladače pro Windows, Mac OS, Linux, Raspberry Pi a další. Rovněž lze na stránkách FTDI nalézt ukázkové použití

a to hned pro několik programovacích jazyků (C++, JAVA, C#, Python a další). FT232RL patří do druhé generace USB převodníků a oproti první generaci disponuje možností přepnutí do režimu Bit Bang. Tento režim umožňuje, vhodným softwarem, přepnutí výstupního RS232 na osmi bitovou sběrnici, jejíž linky se dají libovolně programovat. To umožňuje získat na výstupu obvodu libovolné sekvence dat.

FT232RL zapojení je ponecháno dle doporučení výrobce, pouze je doplněno několika součástkami viz. schema 3.3. Na sedmém bitu je hodinový výstup SCL a na druhém datový výstup SDA. Tyto výstupy jsou připojené ke konektorům na podvozku tak, aby vznikla I2C sběrnice s topologií master-slave. Přičemž master je reprezentován konvertorem FT232RL a slave zařízení tvoří mikrokontroléry Atmega16 umístěné na podvozku.



Obr. 3.3: Schéma zapojení obvodu FT232 pro komunikaci s podvozkem.

Ovládací testovací program

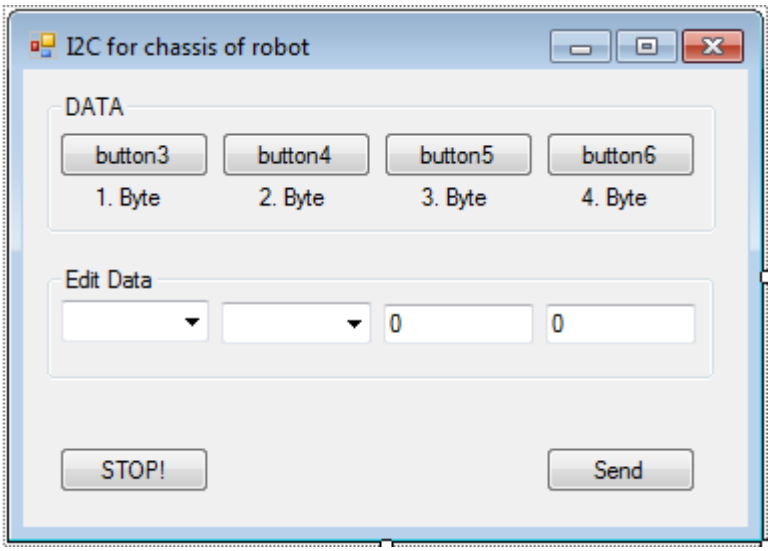
Pro ovládání podvozku byl vytvořen počítačový program, který využívá vlastností konvertoru FT232. Jedná se o jednoduchou formulářovou aplikaci vyvinutou v jazyku C# na platformě .NET4 [3]. Testovací aplikace využívá I2C protokolu podvozku, tudíž lze řídit podvozek přímo z uživatelského počítače.

Součástí aplikace je singleton *FTDI_FT232XX*, který představuje interface mezi

[3]Pro běh této aplikace je nutné mít nainstalovaný .NET4 Framework.

D2XX knihovnou a aplikací. Přes třídu *FTDI_FT232XX* se provádí veškerá komunikace s připojeným obvodem.

Po spuštění a připojení FT232RL[4] k USB, program provede nastavení přenosové rychlosti a Bit Bangu. Pokud některá z těchto procedur selže, program dá o chybě uživateli vědět a nepokračuje v běhu dokud se problém neodstraní. Poté uživatel nastaví patřičná data (pomocí komponent typu) a odešle je na výstup. Tím dojde k inicializaci I2C komunikace a odeslání dat.

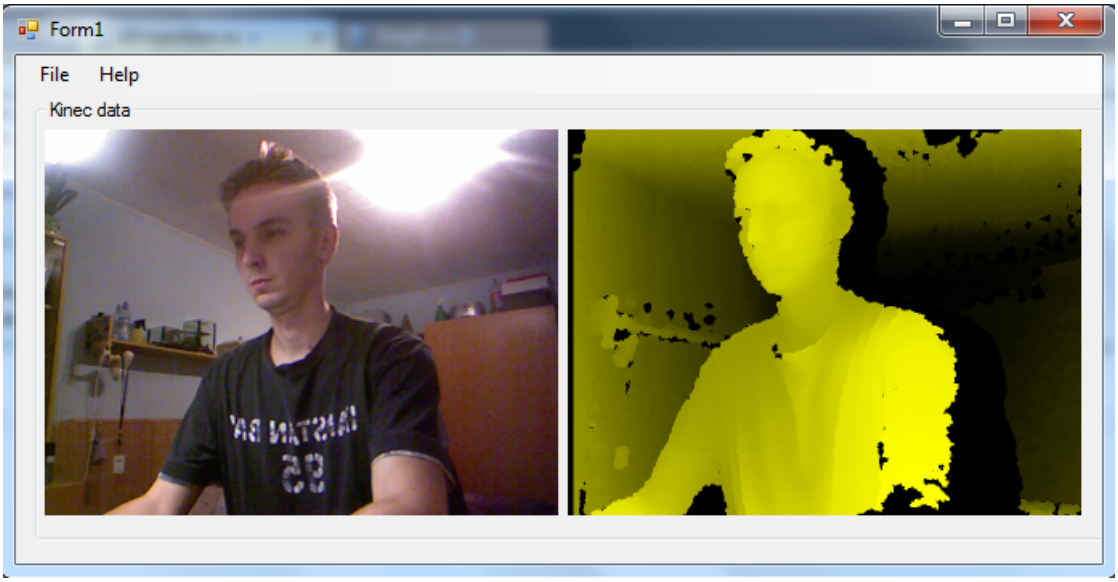


Obr. 3.4: Vytvořený program pro otestování podvozku.

[4]EEPROM obvodu FT232RL musí být nejprve správně nakonfigurovaná. VID = 0x0403 a PID = 0x6001

4 PROGRAM PRO DEMONSTRACI MOŽNOSTÍ ZAŘÍZENÍ MS KINECT

Pro testování možností Kinectu jsem vytvořil program v prostředí Visual Studio. Program má vzhled formulářové aplikace, kde jsou umístěny komponenty pro ovládání programu a sledování informací z Kinectu obrázek 4.1.



Obr. 4.1: Program využívající OpenNI

Nachází se na něm dva panely pro vykreslování RGB a depth dat. Tyto panely jsou objekty odvozené z komponenty *Panel*. Panel je komponenta, ve které je možno shromažďovat různé ovládací prvky a přistupovat k nim z pohledu panelu. V tomto programu panel neplní funkci "shromaždiště komponent", ale využívá se překrytí (override), jeho metody *OnPaintBackground*, která vykreslí Bitmap na pozadí panelu.

4.0.2 Popis vytvořeného programu

Funkce programu je následující: Při spuštění programu proběhne kontrola připojení Kinectu k USB portu počítače. Pokud se připojení nepodařilo, program se ukončí a informuje uživatele o neúspěšném připojení. Je-li však Kinect připojený, dojde k zavolání konstruktoru celé aplikace. V tomto konstruktoru se inicializují proměnné, knihovny, licence OpenNI a základní nastavení Kinectu. Licenci a základní nastavení představují dva XML soubory: *licenses.xml* a *SampleConfig.xml*. V souboru licenses se nachází licenční klíč, bez kterého není možné použít OpenNI [1].

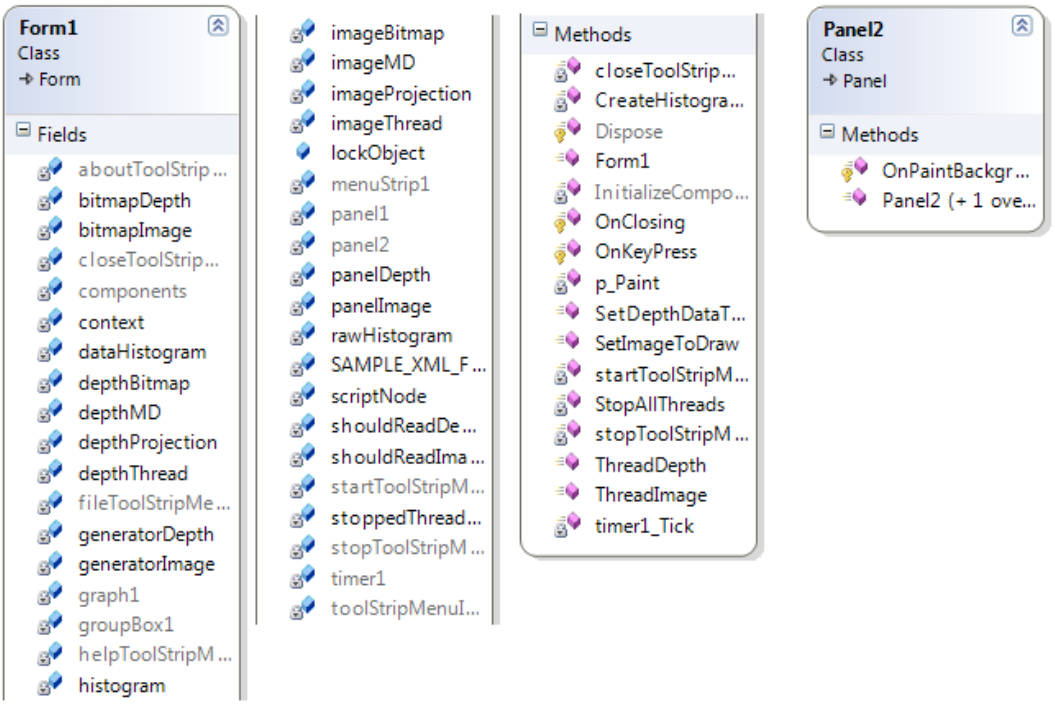
[1] Podle [6] je jinou možností licenci zapsat přímo do zdrojového kódu aplikace.

V souboru *SampleConfig.xml* jsou základní informace jako názvy použitých uzlů, rozlišení výstupu kamery, zrcadlové otočení obrazu a FPS kamery.

Pokud uživatel spustí tlačítkem v záložce *File* program, inicializuje tím dvě vedlejší vlákna: *ImageThread* a *DepthThread*. Tyto vlákna pracují paralelně a mají za úkol získávat příslušná data a zobrazovat je na pozadí panelů.

Vlákno *ImageThread* zpracovává data z objektu *ImageGenerator*, který je součástí OpenNI, a upravuje je do Bitmapového obrázku. Poté tento obrázek voláním metody *Invoke()* vykreslí na pozadí příslušného panelu. Vykreslování se děje takovou rychlostí, že člověk vnímá vykreslený obraz jako video.

Vlákno *DepthThread* pracuje velice podobným způsobem, jen namísto dat z objektu *ImageGenerator* je získává z objektu *DepthGenerator*, který poskytuje informace o vzdálenosti objektů, předmětů před Kinectem. Hodnoty získané z objektu *DepthGenerator* zkalibrují a přiřadí se jim hodnota v rozmezí 0-255. Poté dojde k vytvoření bitmapového obrázku jen s tím rozdílem, že obrázek obsahuje pouze odstíny jedné barvy, nikoliv kompletní RGB. Odstín barvy představuje vzdálenost v tom daném místě obrázku. Následně se zavolá metoda *Invoke()*, která překreslí obrázek na pozadí druhého panelu.



Obr. 4.2: Proměnné a funkce vytvořeného programu.

Ukázka licenčního souboru *Licenses.xml*:

```
<Licenses>
```

```
    <License vendor="PrimeSense" key="0KOIk2JeIBYClPWVnMoRKn5cdY4=" />
</Licenses>
```

4.1 Parametry USB sběrnice Kinectu

Pro získání představy o konfiguraci a interfacu USB sběrnice Kinectu lze použít některou z volně dostupných knihoven- libusb (Linux), LibUsbDotNet (Windows). Tyto knihovny umožňují získat potřebné informace o zapojených zařízeních, čehož lze poté využít při tvorbě komunikačního rozhraní pro USB sběrnici.

4.1.1 Knihovna LibUsbDotNet

LibUsbDotNet je USB knihovna podporující platformy unix-like, Windows. Knihovna je vytvořená v technologii .NET a umožňuje snadnou a rychlou tvorbu USB driverů[10], umožňuje náhled zařízení a dokáže zpracovávat informace jednotlivých deskriptorů zařízení, která jsou připojená k USB sběrnici. LibUsbDotNet byla použita pro získání informací o USB sběrnici kinectu. Tabulka B.1 uvádí informace o připojených zařízeních.

Kamera, audio a motor jsou připojeny k *hubu*. A tím jsou všechna zařízení zapojena na jednu USB sběrnici. V tabulky B.1 jde vidět, že Kinect se skládá ze čtyř USB zařízení:

- Hubu,
- kamery,
- audia,
- motoru.

V tabulce B.2 je uveden deskriptor zařízení motoru, který nastavuje sklon Kinectu. Je vidět, že motor obsahuje jeden konfigurační deskriptor tab. B.3, ze které je patrný počet endpointů zařízení. V tomto případě je počet endpointů 0, což znamená, že zařízení motoru má pouze kontrolní endpoint. Rovněž se z deskriptoru dá zjistit napájení[2] daného zařízení, které je pro motor Kinectu 100mA. V interface deskriptoru kamery tab.B.7 je definovaný počet endpointů -2. Jeden endpoint náleží VGA kameře a druhý Infra kameře Kinectu. Deskriptory těchto endpointů stejné tabulka B.8, se liší pouze v adrese endpointu. První endpoint se nachází na adrese 0x82 a druhý na 0x81. Tyto endpointy používají izochronní komunikaci, tzn. že každých 125 mikrosekund endpointy Kinectu odesílají data do připojeného USB hostu.

[2]Hodnota uvedená v deskriptoru x 2 [mA].

5 BEAGLEBOARD XM

Výběr vhodného zařízení, které by pracovalo dle očekávání, byl velmi složitý. Abych předešel případnému nezdaru, oslovil jsem Drew Fishera z komunity OpenKinect. Výsledkem naší korespondence bylo získání informací o přibližném výkonu ARM procesoru a potřebném hardwaru.

Jednoduchý procesor nemá potřebný výkon k tomu, aby dokázal zpracovávat vysokorychlostní přenosy, které jsou použity u Kinectu. Kinect používá USB2.0 s průměrnou rychlostí 12MB/s pro RGB video a 12MB/s pro zpracování hloubkových dat (depth stream).

K tomu, aby se dala zpracovávat data z Kinectu, musí se každých $125\mu s$ zajistit izochronní přenos, což je velice náročná operace z pohledu kontroly velikosti paketů, hlaviček, kopírování dat do pracovních bufferů a návratů do hlavní smyčky, kterou musí procesor zajišťovat. Velké požadavky jsou rovněž kladeny na velikost paměti RAM. Například velikost jednoho hloubkového framu je $614400B$ což znamená, že RAM by měla mít minimálně třikrát tak větší velikost[1]. To proto, aby dokázala uchovávat předešlý snímek a pracovat s ním, zatímco bude přijímat snímek následující.

Z toho důvodu byl nakonec vybrán minipočítač BeagleBoard-xM Rev.C, který je zapůjčen od vedoucího této práce.

5.1 BeagleBoard-xM Rev.C a jeho vlastnosti

BeagleBoard-xM je miniaturní počítač založený na architektuře ARM. Tento minipočítač disponuje 1GHz procesorem DM3730CBP, který je vyráběn firmou Texas Instruments. Procesor je vytvořený technologií POP (Package on Package), u které je paměťový obvod umístěný na procesoru. Z toho důvodu není možné při pohledu na BeagleBoard xM zahlédnout procesor DM3730CBP, ale lze zahlédnout paměť umístěnou na procesoru. Parametry umístěné paměti jsou 4Gb MDDR SDRAM x32 a je jedinou pamětí, kterou minipočítač má. Avšak toto zařízení umožňuje několik způsobů, jak paměť zvětšit:

1. Použitím paměti mikro SD karty zapojené do mikro SD slotu BeagleBoard-xM,
2. připojením pevného disku k USB OTG,
3. instalováním paměťového zařízení do jednoho z USB portů,

[1] Fisher Drew - osobní korespondence.

4. přidáním redukce z USB na adaptér, který umožňuje připojení pevného disku k USB portu BeagleBoardu-xM.

Napájení minipočítače je možné dvěma způsoby [32]. Přes připojený port USB OTG k osobnímu počítači, který umožňuje omezené napájení BeagleBoardu-xM. Výhodnější ovšem je použití externího napájecího zdroje 5V/2A, který se připojí do napájecího konektoru na BeagleBoardu-xM.

Napájení celého minipočítače řídí obvod TPS65950[32], který disponuje regulátorem, pro napájení dalších periferií BeagleBoardu-xM:

- DVI-D enkoderu,
- RS232 driveru,
- stereo audio výstup/vstup,
- USB OTG,
- signalizační LED a dvě ovládací tlačítka.

Pro vývoj softwaru a debugování je možné k BeagleBoardu připojit 14-pinový JTAG emulátor, který komunikuje se signály o velikosti 1,8V. Podpora RS232 je rovněž součástí. Minipočítač obsahuje RS232 transceiver pro připojení ke kompatibilnímu osobnímu počítači.

BegleBoard xM podporuje několik volně dostupných operačních systémů: Ångström Distribution, Android for BeagleBoard, Ubuntu, Windows Embedded, QNX Neutrino. Z těchto dostupných systému mně nejvíce zaujal svojí podrobně zpracovanou dokumentací a snadným instalováním balíčků operační systém Ångström.

5.2 Bootování operačního systému Ångström

K BeagleBoardu je standardně dodávaná testovací mikro SD karta. Pokud tuto kartu vložíme do mikro SD slotu BeagleBoardu a připojíme napájení celé desky, dojde ke spuštění bootovací sekvence a k zavedení operačního systému Ångström. Po spuštění je možné na obrazovce pozorovat vzhled plně funkčního Ångströmu. Vzhledem k tomu, že tato mikro SD karta je testovací, je nutné vytvořit vlastní bootovací kartu, která by umožňovala experimentování.

Pro spuštění a správnou funkci BeagleBoardu je nutné vytvořit bootovací paměťovou kartu s operačním systémem.

5.2.1 Bootovací SD karta

Bootovací SD karta se skládá ze dvou částí: První část (bootovací) má malou kapacitu a je zformátována ve formátu FAT32. Zbytek SD karty je zformátován ve formátu ext3 a je na ní uložený operační systém Ångström. Vytvoření bootovací karty lze několika způsoby:

- Stáhnutím image karty a následným "vypálením"image na naši SD kartu,
- stáhnutím zabaleného operačního systému včetně bootovacích souborů,
- vytvořením identické kopie z testovací (přiložené) mikro SD karty.

Použití image systému

Image operačního systému je možné stáhnout na oficiálních stránkách Ångströmu. Po stažení image se pomocí programu Disk Imager "vypálí"image systému na naši SD kartu. Nutno podotknout, že tímto způsobem lze vytvořit naši SD kartu pouze na systému Windows.

Zabalený archiv se sytémem

Při vytváření bootovací karty ze zabaleného archivu je postup složitější, obzvláště za použití operačního systému Windows. Je totiž nutné SD kartu rozdělit do dvou částí - FAT32 a ext3. S první částí, která má formát FAT32 si Windows hravě poradí, ale s druhou částí ext3 ne. K tomuto účelu se dá použít program Partition Wizard, nebo jiný, který podporuje formát ext3.

Po vytvoření těchto dvou částí je nutné nakopírovat příslušné soubory na SD kartu. Vzhledem k tomu, že formát ext3 není podporovaný systémem Windows, je potřebné použít program Linux Reader, který dokáže v systému Windows pracovat s tímto formátem. Pomocí tohoto programu se nakopírují soubory pro bootování systému do první části pamětové karty (FAT32) a root systému do části zformátované jako ext3.

Kopie testovací karty

Předešlé způsoby vytváření bootovací SD karty měly hlavní nevýhodu v tom, že nahraný systém obsahoval jen základní balíčky, které umožňovaly pouze práci s konzolí prostřednictvím počítače připojeného k BeagleBoardu. Proto je vhodnější vytvořit image systému z přiložené testovací SD karty, která již obsahuje několik potřebných balíčků. Do SD slotu se vloží testovací SD karta a programem Linux Reader se vytvoří image karty. Tato image se následně programem Disk Imager "vypálí"na námi vloženou SD kartu.

Po vytvoření bootovací SD karty a zasunutí do mikro SD slotu BeagleBoardu se po připojení napájecího napětí rozběhne operační systém Ångström v celé své kráse. V případě zapojeného HDMI[2] kabelu, je možné začít okamžitě na minipočítači pracovat.

5.3 Instalace softwarových balíčků

Instalace balíčků probíhá u systému Ångström různě. Lze použít přenosné paměťové médium s příslušnými balíčky, SSH server společně s osobním počítačem, nebo lze BeagleBoard připojit síťovým kabelem k internetu. Připojení samotného minipočítače k internetu je výhodné v tom, že není kromě BeagleBoardu potřeba žádné další zařízení, a všechny dodatečné balíčky si Ångström při instalaci vyhledá sám. Celý průběh instalace balíčků probíhá písemnou formou v okně *Terminal*.

Instalace probíhá tak, že se nejdříve spustí program *Terminal* a uživatel si na stránkách Ångströmu vyhledá balíčky, které si přeje nainstalovat. Poté zadá do *Terminalu* příkaz: **opkg install "název souboru"** a stiskne klávesu *Enter*. Tím dojde k samotné instalaci daného souboru a všech potřebných balíčků pro správnou funkci instalovaného programu.

5.4 Vývoj aplikací

Pro vývoj aplikací je možné, mezi balíčky Ångströmu nalézt ARM-Linux-gcc kompilátor C/C++, nebo vývojový nástroj Mono pro programování v C#. Další možností je použít komerční verze vývojového prostředí od firmy KEIL: DS-5 Development Tools for ARM Linux and Android. Tímto nástrojem se dají vyvíjet potřebné programy na klasickém počítači a následným překopírováním, prostřednictvím SSH serveru do BeagleBoardu, se spustí. Výhodou DS-5 je přátelštější ovládání a snadná oprava vzniklých chyb. Naopak za nevýhodné považuji složitější nastavení než u GCC kompilátoru a pořizovací cenu.

5.4.1 BeagleBoard - Hello World

Pro demonstraci použití GCC kompilátoru lze použít následující příklad v C++. Nejprve se v operačním systému Ångström otevře textový editor *EDIT*, do kterého se napíše následující kód:

[2]HDMI kabel musí být připojený již před zapojením minipočítače ke zdroji napětí. Kabel nesmí být připojen za běhu, dochází tím ke zničení celého BeagleBoardu-xM [32]

```
#include <iostream>
using namespace std;
int main()
{
  cout<<"Hello World from BeagleBoard-xM" << endl;
  return 0;
}
```

Po zapsání kódu se celý tento dokument uloží s koncovkou .cpp do složky:

```
/home/root
```

Poté stačí v terminálu Ångströmu napsat následující příkaz:

```
arm-angstrom-linux-gnueabi-g++ HelloWorld.cpp -o HelloWorld
```

Tím dojde ke kompilaci vytvořeného programu a vznikne aplikace s názvem HelloWorld. Následujícím příkazem se zkompilovaný program spustí a na obrazovce se v konzoli vypíše "Hello World from BeagleBoard-xM".

```
./HelloWorld
```

5.5 BeagleBoard-xM a MS Kinect

Vzhledem k tomu, že byl BeagleBoard vybrán jako vhodný minipočítač pro komunikaci s Kinectem, další část se věnuje popisu instalace a vlastnostem této konfigurace.

5.5.1 Instalace knihovny libfreenect

Pro práci se zařízením MS Kinect je Ångström obzvláště výhodný, protože mezi jeho balíčky, dostupnými na oficiálních stránkách, se nachází balíček *libfreenect*, který představuje zkompilovanou knihovnu, pocházející z open source projektu OpenKinect. Po instalaci tohoto balíčku, se do systému rozbalí knihovny optimalizované pro operační systém Ångström, včetně jejich hlavičkových souborů. S těmito knihovnami lze poté vyvíjet libovolný software využívající Kinect. Než se však začne používat knihovna *libfreenect*, je nutné nainstalovat balíček lib-usb, který představuje low-level knihovnu pro práci s USB BeagleBoardu.

Instalace libfreenect probíhá následovně:

- opkg install libfreenect
- opkg install libfreenect-dev
- opkg install libfreenect-static

Po skončení instalace libfreenect se nainstaluje lib-usb: opkg install lib-usb

Jakmile se instalace dokončí, je potřeba si uvědomit, že data získaná z Kinectu je možné graficky zpracovávat. Proto je vhodné přiinstalovat balíček OpenCV:

- opkg install OpenCV
- opkg install OpenCV-dev
- opkg install OpenCV-static

Po ukončení instalace je možné vyvíjet software pro Kinect s možností použití knihoven libfreenect a libfreenect-sync. Výkonnost libfreenect knihoven při použití BeagleBoardu-xM je patrná na první pohled po spuštění programu. Nejlépe je na tom synchronní knihovna. Ostatní knihovny jsou na tom daleko hůře. Srovnání výkonů, které byly testovány na minipočítači BeagleBoard-xM za použití operačního systému Ångström, při vykreslování depth a RGB dat je uvedeno v tabulce 5.1.

Tab. 5.1: Tabulka pro porovnání výkonů knihoven libfreenect použitých systémem Ångström.

Název knihovny	**obrázků/s**	**Subjektivní hodnocení**
libfreenect-sync.h	6	nepatrné zpoždění
libfreenect.h	méně než 1	je patrné zpoždění přibližně 2 sekundy
libfreenect.hpp	méně než 1	je patrné zpoždění přibližně 2 sekundy

5.5.2 Open Computer Vision - OpenCV

OpenCV (Open Computer Vision) je otevřená multiplatformní knihovna pro manipulaci s obrazem, vyvinutá firmou Intel. Zaměřuje se především na počítačové vidění a zpracování obrazu v reálném čase. Knihovnu je možné použít v prostředí jazyku C a C++, čehož je využito pro vývoj na minipočítači BeagleBorad. Nejprve se celé OpenCV, včetně podpůrných balíčků, musí nainstalovat do systému Ångström:

opkg install OpenCV

Pro představu práce s OpenCV je zde uvedena ukázková aplikace, která načte a zobrazí obrázek uložený v adresáři aplikace:

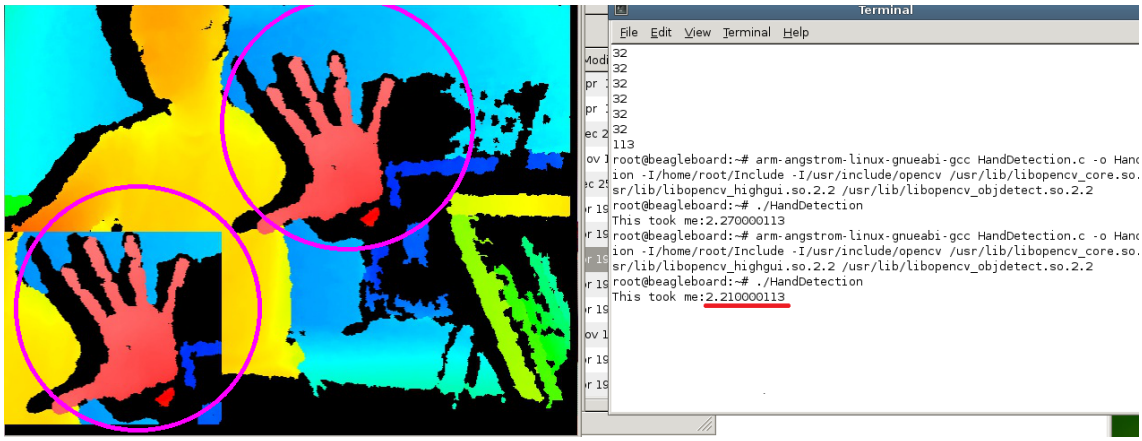
```
#include <stdio.h>
#include <stdlib.h>
#include <cv.h>
#include <highhui.h>

char * windowName="OpenCV sample";
IplImage * img=NULL;
int main()
{
  img=cvLoadImage("Obrazek.jpg",1);
  cvNamedWindow(windowName,CV_WINDOW_AUTOSIZE);
  while(cvWaitKey(20)!='e')
  {
  }
  cvReleaseImage(&img);
  return 0;
}
```

Velkou výhodou OpenCV je možnost použít tzv. HAAR-like features. Jsou to digitální obrazové prvky, vycházející z Haarové vlnky, využívané při rozpoznávání objektů. Haar-like features byly poprvé použity v real-time obličejovém detektoru. Umožňují však rozpoznávat daleko více než jen obličej, např. ruce, lidské tělo a další.

V praxi je Haar prvek reprezentován souborem obsahujícím konkrétní data potřebná pro detekování daného objektu. Tento soubor se v programu načte a pomocí vhodných funkcí využije.

OpenCV: HAAR-like features

Původní vize, která byla zamýšlena, počítala s použitím HAAR prvku z OpenCV. Při testování této možnosti vznikl problém s výkonem minipočítače. Byl vytvořen program, ve kterém měl BeagleBoard-xM za úkol detekovat dvě ruce ve statickém snímku. Program pracoval spolehlivě, ale čas vyhrazený pro detekci, byl nesmírně vysoký. Doba detekce rukou činila přes 2 sekundy, což bylo pro výsledného robota nepřijatelné. Pro inspiraci je na obrázku 5.1 zobrazena detekce rukou a výsledný čas.



Obr. 5.1: Detekce rukou v obrázku.

5.6 BeagleBoard a FTDI driver

Instalace podpůrného rozhraní pro komunikaci se sériovým modulem založeným na obvodu FT232 již není tak snadné. Nejprve je nutné si stáhnout patřičné balíčky a vytvořit si vlastní knihovnu pro FTDI obvody dle následujícího postupu:

- libftdi - balíček obsahuje zdrojový kód driveru pro FTDI obvody. Je dostupný na webových stránkách *http://www.intra2net.com/en/developer/libftdi/download.php*.
- libusb-compat- tento balíček obsahuje USB knihovnu kompatibilní s FTDI driverem. Bez této knihovny není možné zdrojové soubory z libftdi použít. V tomto případě se musí jednat o knihovnu s popisem compat, která definuje proměnné kompatibilní s libftdi balíčkem. Tento balíček je volně ke stažení na oficiálních stránkách Ångströmu.

Nejprve se rozbalí libusb-compac, čímž vzniknou dva další balíčky (DATA a INKJ), které je nutno také rozbalit. Po rozbalení balíčku DATA.psk jsou ve složce *src/lib* dostupné knihovny a v *src/include* hlavičkový soubor k těmto knihovnám.

Všechny soubory se nakopírují do složky Ångströmu: */usr/lib* a hlavičkový soubor do */src/include*. Po nakopírování všech souborů je možné knihovnu libovolně přilinkovat.

Jak již bylo zmíněno výše, součástí libftdi balíčku jsou zdrojový (.c) a hlavičkový (.h) soubor. Aby se tento balíček dal dále využít, je vhodné z něj vytvořit tzv "shared"knihovnu. Tím dojde ke značnému usnadnění při psaní příkazu pro kompilátor a snadnější správě kódu (ve výsledném programu se pouze "includuje"hlavičkový soubor a při kompilaci se přilinkuje daná knihovna).

Samotná knihovna se vytvoří následujícím příkazem:

```
arm-angstrom-linux-gnueabi-g++ -shared libftdi.c -o libFTDI.so
-I/usr/src/include/libusb /usr/src/lib/libusb.so
```

5.6.1 Program pro sériový modul

Jedná se o program, běžící na operačním systému Ångström, který komunikuje se sériovým modulem, využívá výše zmíněné knihovny libFTDI.so a I2C protokol podvozku. Nejprve program definuje adresy slave zařízení, výstupní piny (SDA a SCL) na výstupu obvodu FT232RL a jednotlivé povely pro dané slave zařízení. Tyto povely jsou vlastně připravené funkce pro odesílání příkazů, prostřednictvím I2C, definovaných v tabulce 3.1. Funkce pro pohyb podvozku jsou shrnuty v příloze G.1:

6 VYTVOŘENÝ MOBILNÍ ROBOT

Mobilní robot je seskládán z dříve popsaných částí. Na rám podvozku jsou, uprostřed pomocí šroubků, připevněny ploché díly získané ze známé stavebnice Merkur. Na těchto ploškách je připevněn MS Kinect, LCD displej, BeagleBoard-XM a chladič pěti voltového stabilizátoru. K připevněným dílům jsou, pomocí vázacích drátů, připevněné napájecí a USB kabely všech částí.

Obr. 6.1: Vytvořený mobilní robot.

6.1 Hardware, jeho nastavení a zabezpečení

6.1.1 Umístění minipočítače

Aby nedošlo k poškození minipočítače, ať už při jízdě, nebo manipulaci, je umístěn v černé plastové krabičce, jejíž výkres se nachází v příloze E.1. V krabičce jsou vybroušené otvory pro USB, S-video, HDMI, zvukový vstup a výstup, a nakonec i konektor pro napájení. Provedení krabička však neumožňuje manipulaci s mikro SD kartou[1] a připojení k sériovému portu. Tato krabička je poté připevněna,

[1]Při neopatrném zapojování zařízení do USB portů BeagleBoardu-xM, může vlivem nevhodného umístění SD slotu, dojít ke stisknutí mikro SD karty. Pokud k tomuto dojde při běhu systému,

na konstrukci robota, hned za MS Kinectem.

6.1.2 Sériový modul

Sériový modul využívá upraveného zapojení obvodu FT232RL z obrázku 3.3. Dále tvoří základní napájecí desku, ze které se rozvádí veškerá elektrická napětí do jednotlivých částí robota. Zapojení je patrné ze schématu v příloze D. Na dvanácti voltový vstup je připojen napájecí zdroj CPS-S 12/10A, ze kterého se získává 5V/2A k napájení BeagleBoardu xM a obou procesorů slave zařízení. Pro napájení motorů robota je na Sériovém modulu vyvedena zvláštní svorka, která může sloužit jako interface pro připojení dalších modulů. Na desce nechybí ani napájecí napětí pro displej a Kinect, který se k napájení připojuje přes upravený Kinect konektor. Aby nedošlo k poškození celého zařízení během pohybu, je celá deska upevněna, pomocí šroubů, v části krabičky, jejíž parametry jsou uvedeny v příloze E.2.

Stabilizátor IC2, který poskytuje 5V napájení pro BeagleBoard, není umístěn přímo na DPS sériového modulu, ale mimo něj na hliníkovém chladiči o rozměrech 100x100x10. Chladič se stabilizátorem se nachází v zadní části robota, kde je stabilizátor dostatečně chlazen a nedochází tak k přerušení funkce vlivem tepelné ochrany.

6.1.3 LCD displej a jeho zprovoznění

Jako displej jsem použil 7 palcový TFT monitor, s rozlišením 800x480 pixelů, určený pro sledování videa v automobilu. Tento LCD monitor jsem vybral proto, že umožňuje přes redukci připojit ke kompozitnímu výstupu videa BeagleBoardu.

Připojení je potřeba aktivovat, v souboru *uEnv.txt*, kompozitní výstup:

```
mpurate=1000
dvimode="hd720 omapfb.vram=0:8M,1:4M,2:4M"
vram=16M
optargs="consoleblank=0"
console="tty0 console=ttyS2,115200n8"
```

Na druhý řádek souboru *uEnv.txt* se vloží formule *defaultdisplay=tv* namísto *dvimode=”hd720 omapfb.vram=0:8M,1:4M,2:4M”*. Aby vše proběhlo v pořádku, musí se soubor uložit a uživatel musí provést odhlášení (Log Out)[2] ze systému. Poté se znovu přihlásit a zkontrolovat správnost uložených dat a nakonec resetovat.

dojde k narušení integrity paměťové karty a porušení celé image systému.

[2]Samotné uložení souboru nestačí - proto pokud uživatel tento krok neprovede, nedojde k připsání *defultdisplay=tv* do souboru. Po resetu se BeagleBoard-xM spustí obvyklým způsobem bez zapnutého video výstupu.

Po následujícím resetu, během přihlášení, již dojde k zapnutí kompozitního výstupu. Druhou možností je přepsání těchto hodnot na uživatelském počítači při vložené mikro SD kartě.

Tímto způsobem se dosáhne spuštění výstupu kompozitního videa a vypnutí HDMI výstupu. Obdobným způsobem lze nastavit BeagleBoard-xM pro DVI-D výstup, musí se jen změnit hodnota druhého řádku: *defaultdisplay=dvi*. Pro změnu rozlišení DVI výstupu je možné použít dostupné formáty v [31].

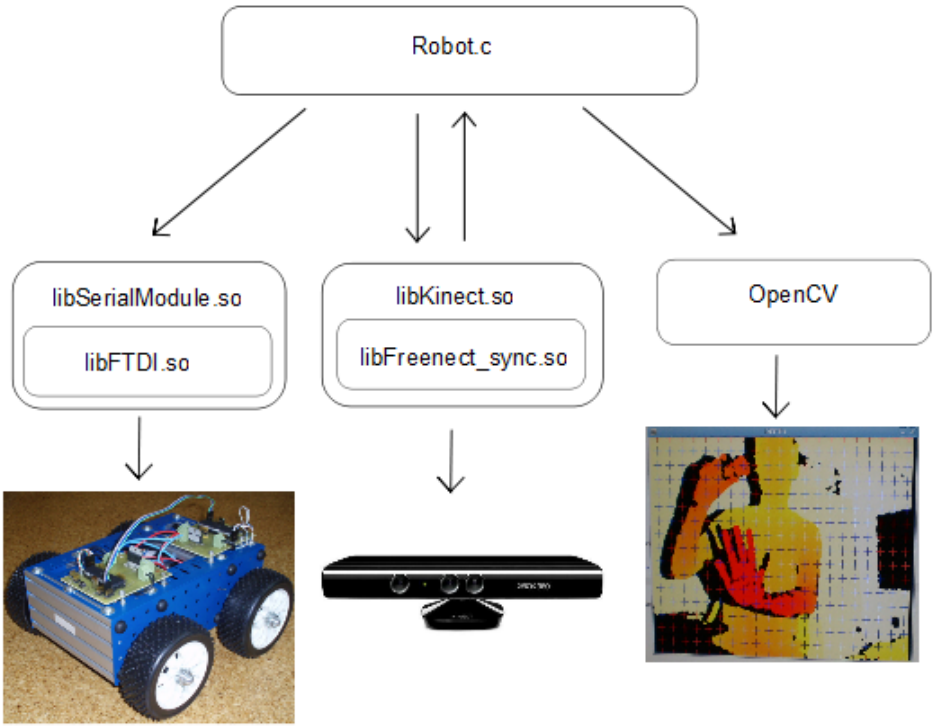
6.2 Softwarová část robota

Při vývoji softwaru bylo postupováno tak, aby se nejednalo o jeden dlouhý a nepřehledný kód. Byl kladen důraz na přehlednost a možná další rozšíření. Celý program robota byl naprogramován v jazyku C a zkompilován přímo na minipočítači. Struktura softwaru je patrná z obrázku 6.2, který se nachází na následující straně. Jádro programu tvoří zdrojový soubor *Robot.c*, který spolupracuje se třemi knihovnami:

1. libSerialModule - komunikuje s podvozkem prostřednictvím I2C sběrnice,
2. libKinect - vhodně zpracovává data z Kinectu,
3. OpenCV - slouží vykreslování informací a dat.

Pro představu principu funkčnosti robota je níže uvedena ukázka zdrojového kódu pro volání základních funkcí:

```
void RobotInit(){...}
int Stop(){...}
void Video(float time){...}
void Depth(float time){...}
.
// Update video a depth dat v main()
while(cvWaitKey(10)<0)
{
  t=clock();
  Video(((float)t)/CLOCKS_PER_SEC);
  Depth(((float)t)/CLOCKS_PER_SEC);
  t=clock()-t;
}
```



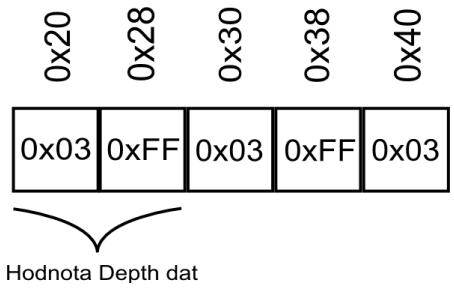
Obr. 6.2: Struktura softwaru robota

Po spuštění programu, se načtou proměnné samotného robota a po jejich načtení se zavolá funkce *Init()*, která inicializuje všechny části robota - Kinect, sériový modul a proměnné potřebné pro zobrazení dat pomocí OpenCV. Proběhne-li tato procedura úspěšně, přejde program do nekonečné smyčky ve funkci *Update*. V této funkci se volají, zpracovávají a vykreslují patřičná data na monitor robota. Robot vypočítává vzdálenosti některých objektů a na základě vyhodnocených dat se rozhoduje, zda má odeslat povel pro jízdu vpřed, či nikoliv.

6.2.1 Zpracování vzdálenosti

Zpracování vzdálenosti má na starosti knihovna *libKinect.so*, která obsahuje základní strukturu, funkce pro inicializaci a deinicializaci. Kromě toho se v této knihovně nachází privátní funkce, které jsou potenciálnímu developerovi, nepřístupné. Tyto privátní funkce mají za úkol přijmout přijatá data Kinectu, zpracovat, a vhodně je v paměti uschovat. Tento proces probíhá tak, že se nejprve do bufferu načtou veškerá 11-bitová depth data a uloží se do struktury IplImage, která obsahuje pole typu char - imageData. Vzhledem k tomu, že imageData jsou typu char a data potřebná pro uložení, jsou 11-bitová, musí se při inicializaci struktury IplImage vytvořit dvakrát tak větší místo v paměti, aby se do struktury všechna data vešla.

Umístění dat v paměti bude odpovídat případu uvedenému na obrázku č.6.3.



Obr. 6.3: Způsob uložení dat v paměti.

6.3 Standardní pohyb robota po prostoru

Jedná se o jeden z pohybů mobilního robota. Logiku tvoří program, podobný stavovému automatu, který získává a zpracovává informace z Kinectu.

Po spuštění programu robot 5 sekund v klidu čeká. Po této době začne vyhodnocovat vzdálenosti objektů, vyskytujících se před ním. Pokud se před robotem nenachází překážka, je tam volný prostor, vydá se robot vpřed. Pokud narazí na překážku, vyhodnotí prostor po pravé a levé straně před robotem, změří jejich vzdálenost a vydá povel k rotaci ve směru v kterém je překážka více vzdálená od robota.

Vzhledem k tomu, že technologie Kinectu není dokonalá (viz. kapitola "Hlavní nevýhoda zařízení"), dochází k chybám při vyhodnocování dat. Tuto vlastnost bylo nutné omezit tak, aby nezpůsobovala robotovi vážné potíže při pohybu. Při přibližování k předmětům se stává, že senzor Kinectu vyhodnotí vzdálenost objektu a vedle něj jeho stín. Tento stín nabývá nesprávné hodnoty, která je stejná jako hodnota hloubkových dat pro objekty blíže než 50 cm. V tomto okamžiku si Kinect "myslí", že je v trase překážka a snaží se jí vyhnout.

6.3.1 Kontrola depth dat

Tento problém byl vyřešen pomocí algoritmu, který před vyhodnocením dat, data validuje a tím do určité míry potlačí nevyžádaný pohyb robota.

Vzhledem k tomu, že BeagleBoard není žádný "super"počítač, a vyhodnocování veškerých depth dat by procesor příliš zatížilo, program zpracovává pouze některá data. Tato zpracovávaná data představují průsečíky křížků (viz obrázek). Jakmile jsou dostupná hloubková data, algoritmus zpracuje hodnoty dat v těchto bodech. Po ověření dat se buď robot rozjede/pokračuje v jízdě, nebo začne hledat jinou dostupnou cestu.

6.3.2 Rozhodování robota - stavový automat

Robot se rozhoduje na základě výsledků získaných z Depth dat. Aby jeho pohyb nebyl náhodný, měl nějaký řád, byl program doplněn stavovým automatem. Tento stavový automat představuje do určité míry rozhodovací interface mezi vyhodnocenými daty a povely pro podvozek. Na základě povelů a stavů se automat přepíná a tím řídí pohyb podvozku robota.

6.4 Povelem řízený robot

Druhý režim robota se snaží, alespoň z části, napodobit detekci člověka a na základě poloh jeho rukou vyhodnotit povel pohybu robota. Kolem detekované osoby se nachází čtyři tlačítka, která slouží jako spínače pro nastavený povel. Jsou to vlastně call-back funkce nastavené z hlavního programu. Takto definovaný detektor neumožňuje developerovi možnost zasahovat do funkce detektoru, ale umožňuje nastavení různých povelů pomocí těchto call-back funkcí.

Princip je následující:

1. Po spuštění aplikace dojde ke spuštění všech částí robota a v programu se nastaví vyhodnocovací *Treshold*, který detekuje objekty v rozsahu 2-3 metrů od robota,
2. určená osoba se postaví před robota s rozpaženýma rukama,
3. tím dojde v programu k detekci postavy na základě počtu bodů ve vertikální a horizontální rovině,
4. pokud osoba provede "stisknutí tlačítka", dojde k vykonání povelu.

Pro demonstraci bylo naprogramováno chování robota, které při zvednutí levé ruky dá povel robotu pro jízdu vzad. Při zvednutí pravá ruky dojde k udání povelu vpřed. Při pohybu vpřed, či vzad se musí uživatel pohybovat společně s robotem, aby nedošlo ke ztrátě kontaktu mezi robotem a uživatelem. Pokud nastane situace, kdy se uživatel ocitne mimo hranici detekce, dojde k zastavení robota a přepnutí na opětovnou kalibraci. Jakmile uživatele robot znovu detekuje, lze pokračovat v ovládání.

7 ZÁVĚR

Cílem této práce bylo seznámit se s vlastnostmi a možnostmi zařízení MS Kinect a tyto vědomosti dále předat a patřičně využít. Seznámit se s různými softwary podporujícími práci s Kinectem. Definovat potřebný hardware a sestrojit mobilního robota, který by využíval těchto vlastností k jednoduchému pohybu po prostoru.

Pozornost byla především věnována knihovně libfreenect a frameworku openNI. Microsoft Windows SDK a CLNUI nebyly v této práci testovány, nýbrž jen zmíněny. Jako řídící jednotka byl zvolen minipočítač BeagleBoard-xM Rev.C, který disponuje dostatečným výkonem pro práci s Kinectem. Práce popisuje vytvoření bootovací SD karty s operačním systémem Ångström a prakticky se snaží nastínit princip vývoje softwaru.

V konečné fázi je práce zaměřena na popis jednotlivých částí robota až po jeho finální podobu. Vytvořené zařízení se umí, na základě depth dat z Kinectu, autonomně pohybovat po prostoru bez toho aniž by přišlo do kontaktu s překážkou. Při pohybu jsou zpracovávaná a vykreslovávaná pouze depth data. Vykreslování RGB dat bylo vynecháno k vůli úspoře výkonu minipočítače. S touto konfigurací je dosaženo rychlosti vykreslování dat až 12 FPS.

Pro zpracování povelů byly nejprve vyzkoušeny možnosti OpenCV - Haar. Tato cesta se však nejevila jako ta pravá. Zpracování jednoho snímku v tomto případě trvalo přibližně 2 sekundy, což se negativně projevilo na pohybu robota.

Výsledkem této práce je především funkční robot, několik vytvořených programů pro běžný počítač, BeagleBoard-xM a poznatky některých již zkušených uživatelů Kinectu a BeagleBoardu-xM.

Robot umožňuje pohyb mezi překážkami dostatečné velikosti. Dovoluje omezený pohyb pomocí povelů. Výsledný kód je snadné využít samostatně pro otestování funkčnosti jednotlivých částí robota, nebo jako základ pro další budoucí projekty.

7.1 Nedostatky robota, námět na vylepšení

Tento funkční robot má stejně jako jiná špičková zařízení množství vlastností, které se negativně projevují na jeho funkčnosti. Jednou z takových vlastností je nevhodná volba převodového poměru všech kol. Její poměr činí 67,5:1, který způsobuje, že robot je při jízdě dosti svižný - alespoň při pohybu vpřed a vzad při maximálních otáčkách. Tento rychlý pohyb se s uvedenou konfigurací jeví jako jediný možný. Při snížení otáček motorů dochází k zablokování všech kol vlivem hmotnosti robota. Řešením by bylo nahradit převodovku jinou s nižším převodním poměrem - snížila by se tím sice maximální rychlost, ale zvýšila nosnost podvozku.

Pohybuje-li se robot úzkou chodbou, na jejíž konci se nachází odbočka, může dojít k přehlédnutí této odbočky vlivem. Robot dorazí na konec chodby, rozhodne se hledat alternativní cestu, avšak rychlost rotace podvozku je tak vysoká, že robot zpracuje hloubková data mimo odbočku a začne se točit na místě. Řešením takovéhoto problému je uzpůsobit rozměry prostoru, ve kterém se robot pohybuje, nebo zařídit pomalejší pohyb robota při rotaci.

LITERATURA

[1] BUNDALO, V. *A look back at Kinect´s history*[online]. 2011-[cit. 8. 4. 2011]. Dostupný z WWW: <http://www.webestigate.com/2011/02/23/a-look-back-at-kinects-history/>.

[2] JAVŮREK, K. *Microsoft Kinect: nová éra, tělo jako ovladač*[online]. 2010-[cit. 8. 4. 2011]. Dostupný z WWW: <http://www.zive.cz/clanky/microsoft-kinect-nova-era-telo-jako-ovladac/sc-3-a-154556/default.aspx>.

[3] *Kinect sensor*[online]. 2010-[cit. 8. 4. 2011]. Dostupný z WWW: <http://msdn.microsoft.com/en-us/library/hh438998.aspx>.

[4] *Prime Sense*[online]. [cit. 20. 4. 2011]. Dostupný z WWW: <http://en.wikipedia.org/wiki/PrimeSense>.

[5] *OpenNI*[online]. [cit. 20. 4. 2011]. Dostupný z WWW: <http://en.wikipedia.org/wiki/OpenNI>.

[6] DAVISON, A. *Kinect Chapters 1 and 2. Kinect Imaging*[online]. 2010-[cit. 8. 2. 2012]. Dostupný z WWW: <http://fivedots.coe.psu.ac.th/ ad/jg/nui13/index.html>.

[7] *OpenKinect*[online]. 2010-[cit. 8. 2. 2012]. Dostupný z WWW: <http://openkinect.org/wiki/Main_Page>.

[8] *libusb*[online]. 2010-[cit. 8. 2. 2012]. Dostupný z WWW: <http://libusb.org/>.

[9] *DIY Kinect Hacking*[online]. 2010-[cit. 8. 4. 2011]. Dostupný z WWW: <http://www.ladyada.netlearndiykinect>.

[10] *LibUsbDotNet 2.2.8*[online]. 2010-[cit. 8. 4. 2011]. Dostupný z WWW: <http://libusbdotnet.sourceforge.netV2Index.html>.

[11] *BeagleBoard-xM Product Details*[online]. 2010-[cit. 8. 4. 2012]. Dostupný z WWW: <http://beagleboard.org/hardware-xM>.

[12] STRUHELKA, M. *Ovládání motorů po sběrnici:bakalářská práce.*[online]. BRNO: VUT Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky, 2012. 47 s.

[13] *Structured-light 3D scanner*[online]. [cit. 22. 8. 2012]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Structured-light_3D_scanner>.

[14] MELGAR, E. R.,DIEZ, C. C. *Arduino and Kinect Projects* New York: Technology in action, 2012. 393 s. ISBN 978-1-4302-4167-6.

[15] *Asus Xtion: Pohybové ovládání pro PC pochází ze stejné dílny jako Kinect a míří na český trh*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://cdr.cz/clanek/pohybove-ovladani-asus-xtion-pro-predstaveni-parametry-cena>.

[16] ČÍŽEK, J. *Video: Vyzkoušeli jsme Kinect for Windows*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://www.zive.cz/clanky/video-vyzkouseli-jsme-kinect-for-windows/sc-3-a-164250/default.aspx>.

[17] *KINECT for Windows - What´s new*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://www.microsoft.com/en-us/kinectforwindows/Develop/New.aspx>.

[18] *Leap Motion: 200x přesnější bezdotykové ovládání než Kinect [video]*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://www.zive.cz/bleskovky/leap-motion-200-presnejsi-bezdotykove-ovladani-nez-kinect-video/sc-4-a-163775/default.aspx>.

[19] *Documentation: The OpenNI Grabber Framework in PCL*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://www.pointclouds.org/documentation/tutorials/openni_grabber.php#openni-grabber>.

[20] *Look Out Kinect, Here Comes i-dong*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://www.1up.com/news/kinect-i-dong>.

[21] VIKTORIN, R. *Kinect i Move se dočkalo konkurence*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://www.zing.cz/novinky/2111/kinect-i-move-se-dockalo-konkurence>.

[22] *Nastavení systému Kinect*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://support.xbox.com/cs-CZ/xbox-360/kinect/kinect-sensor-setup>.

[23] *Microsoft Kinect má velkou konkurenci. Půjde připojit k televizi i k počítači*[online]. [cit. 22. 11. 2012]. Dostupný z WWW: <http://technet.idnes.cz/microsoft-kinect-ma-velkou-konkurenci-pujde-pripojit-k-televizi-i-k-pocitaci-1w1-/hardware.aspx?c=A110107_204909_hardware_kuz>.

[24] *China High Tech Fair i-Dong Motion Controller*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://webnalyzer.com/china-high-tech-fair-i-dong-motion-controller/>.

[25] *Čínská obdoba Kinectu*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://bonusweb.idnes.cz/cinska-obdoba-kinectu-0t3-/Zurnal.aspx?c=A101121_092640_bw-zurnal-ostatni_vdp>.

[26] *Quadotor+Kinect*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://hybrid.eecs.berkeley.edu/~bouffard/kinect.html>.

[27] VINKLER, M. *Využití pohybového snímače Kinect ve virtuální realitě*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://is.muni.cz/th/208036/fi_m/DP_text.pdf>.

[28] *Microsoft Kinect SDK for Windows In Beta*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://www.funnyrobotics.com/2011/06/microsoft-kinect-sdk-for-windows-in.html>.

[29] *Virtopsy Using Kinect for "Minimally Invasive"Autopsies*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://www.dashhacks.com/kinect/kinect-news/virtopsy-using-kinect-for-minimally-invasive-autopsies.html>.

[30] VIAGER, M. *Analysis of Kinect for mobile robots*[online]. [cit. 22. 11. 2012]. Dostupný z WWW:
<http://www.scribd.com/doc/56470872/3/Light-Coding-with-the-Kinect>.

[31] *How to change the display resolution*[online]. [cit. 22. 1. 2013]. Dostupný z WWW:
<http://www2.sakoman.com/OMAP/how-to-change-the-display-resolution.html>.

[32] *BeagleBoard-xM Rev C System Reference Manual*[online]. [cit. 18. 1. 2013]. Dostupný z WWW:
<http://beagleboard.org/static/BBxMSRM_latest.pdf>.

[33] *Patent application title: DEPTH MAPPING USING PROJECTED PATTERNS*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://www.faqs.org/patents/app/20100118123>.

[34] MACCORMICK, J. *How does the Kinect work?*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://users.dickinson.edu/ jmac/selected-talks/kinect.pdf>.

[35] *Snail Instruments - internetový obchod*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://shop.snailinstruments.com/>.

[36] *OUR FULL 3D SENSING SOLUTION*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://www.primesense.com/solutions/technology/>.

[37] *Kinect Hacking 101: Hack a Powershot A540 for Infrared Sensitivity.*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://www.futurepicture.org/?p=97>.

[38] *FAQ.*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://openkinect.org/wiki/FAQ#What_is_the_wavelength_of_the_laser_illuminator_and_what_about_the_dot_patten_used.3F>.

[39] *ASUS Xtion PRO*[online]. [cit. 1. 1. 2013]. Dostupný z WWW:
<http://www.czc.cz/asus-xtion-pro/89488/produkt>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

USB USB - Universal Serial Bus - Univerzální sériové rozhraní

HUB rozbočovač, ke kterému jsou připojené další zařízení

Slave podřízené zařízení

.NET .NET - dot NET. Soubor technologií, které tvoří celou platformu, dostupnou pro Windows.

C# C Sharp. Programovací jazyk vyvinutý firmou Microsoft.

API Application Programming Interface

OpenCV Open Computer Vision - volné knihovny pro počítačové vidění

OpenNI Open Natural Interaction

I2C Inter-Integrated Circuit - multimasterová sběrnice

SDA I^2C datový signál

SCL I^2C hodinový signál

PWM Pulse Width Modulation - pulzně šířková modulace

FPS Frames Per Second - jednotka snímkové frekvence

TOF Time Of Fly - Doba letu. Technika používaná při měření vzdálenosti. Využívá měření doby od vyslání po přijetí signálu, ze které určí vzdálenost.

IR Infrared - Infračervené záření. Elektromagnetické záření s vlnovou délkou větší než viditelné světlo.

SLS Structured Light Scanning - Strukturované světelné skenování.

ARM Advanced RISC Machine - Architektura procesorů vyvinutá v Británii firmou ARM Limited.

SEZNAM PŘÍLOH

A PARAMETRY BEAGLEBOARDU-XM

Tab. A.1: Seznam vlastností BeagleBoardu-xM Rev.C

Specifikace	**Doporučená hodnota**	**Jednotka**
Procesor	DM3730	-
Kmitočet procesoru	1	GHz
Kmitočet DSP	800	MHz
DDR	512	MB
Vstupní napájecí napětí USB	5	V
Vstupní napájecí proud USB	350	mA
Vstupní napájecí napětí DC adapteru)	5	V
Vstupní napájecí proud DC adapteru)	750	mA
High Speed Mode	480	Mb/s
Full Speed Mode	12,5	Mb/s
Low Speed Mode	1,5	Mb/s

B PARAMETRY USB SBĚRNICE KINECTU

Název	Napájení [mA]	Rychlost [Mbs]	Product ID	Vendor ID
Hub	500	480	0x005a	0x0409
Xbox NUI Camera	500	480	0x02ae	0x045e
Xbox NUI Audio	500	480	0x02ad	0x045e
Xbox NUI Motor	500	12	0x02b0	0x045e

Tab. B.1: Zařízení, interface a konfigurace dostupná prostřednictvím USB [9]

Název	**Hodnota**
DescriptorType	Device
BcdUsb	0x0200
Class	PerInterface
SubClass	0x00
Protocol	0x00
MaxPacketSize0	64
VendorID	0x045E
ProductID	0x02B0
BcdDevice	0x0105
ManufacturerStringIndex	1
ProductStringIndex	2
SerialStringIndex	0
ConfigurationCount	1
ManufacturerString	Microsoft
ProductString	Xbox NUI Motor
SerialString	-

Tab. B.2: Tabulka hodnot device deskriptoru pro zařízení Xbox NUI Motor

Kamera Kinectu obsahuje kromě zařízení, konfiguračního a interface deskriptoru ještě dva endpointové deskriptory.

Název	**Hodnota**
ProductString	Xbox NUI Motor
SerialString	-
Length	9
DescriptorType	Configurate
TotalLength	18
InterfaceCount	1
ConfigID	1
StringIndex	0
Attributes	0xC0
MaxPower	50
ConfigString	-

Tab. B.3: Tabulka hodnot konfiguračního deskriptoru pro zařízení Xbox NUI Motor

Název	**Hodnota**
Length	9
DescriptorType	Interface
InterfaceID	0
AlternateID	0
EndpointCount	0
Class	VendorSpecific
SubClass	0x00
Protocol	0x00
StringIndex	0
InterfaceString	-

Tab. B.4: Tabulka hodnot interface deskriptoru pro zařízení Xbox NUI Motor

Název	**Hodnota**
Length	19 [1]
DescriptorType	Device
BcdUsb	0x0200
Class	PerInterface
SubClass	0x00
Protocol	0x00
MaxPacketSize0	64
VendorID	0x045E
ProductID	0x02AE
BcdDevice	0x010B
ManufacturerStringIndex	2
ProductStringIndex	1
SerialStringIndex	3
ConfigurationCount	1
ManufacturerString	Microsoft
ProductString	NUI Camera
SerialString	A00364914775046A

Tab. B.5: Tabulka hodnot deskriptoru zařízení pro zařízení Xbox NUI Camera

Název	**Hodnota**
Length	9
DescriptorType	Configuration
TotalLength	32
InterfaceCount	1
ConfigID	1
StringIndex	0
Attributes	0xC0
MaxPower	8
ConfigureString	-

Tab. B.6: Tabulka hodnot konfiguračního deskriptoru pro zařízení Xbox NUI Camera

Název	**Hodnota**
Length	9
DescriptorType	Interface
InterfaceID	0
AlternateID	0
EndpointCount	2
Class	VendorSpecific
SubClass	0xFF
Protocol	0xFF
StringIndex	0
InterfaceString	-

Tab. B.7: Tabulka hodnot interface deskriptoru pro zařízení Xbox NUI Camera

Název	**Hodnota**
Length	7
DescriptorType	Endpoint
EndpointID	0x81 [2]
Attributes	0x01
TransferType	Isochronous[9]
MaxPacketSize	3008[3]
Interval	1
Refresh	0
SynchAdress	0x00

Tab. B.8: Tabulka hodnot endpoint deskriptoru pro zařízení Xbox NUI Camera

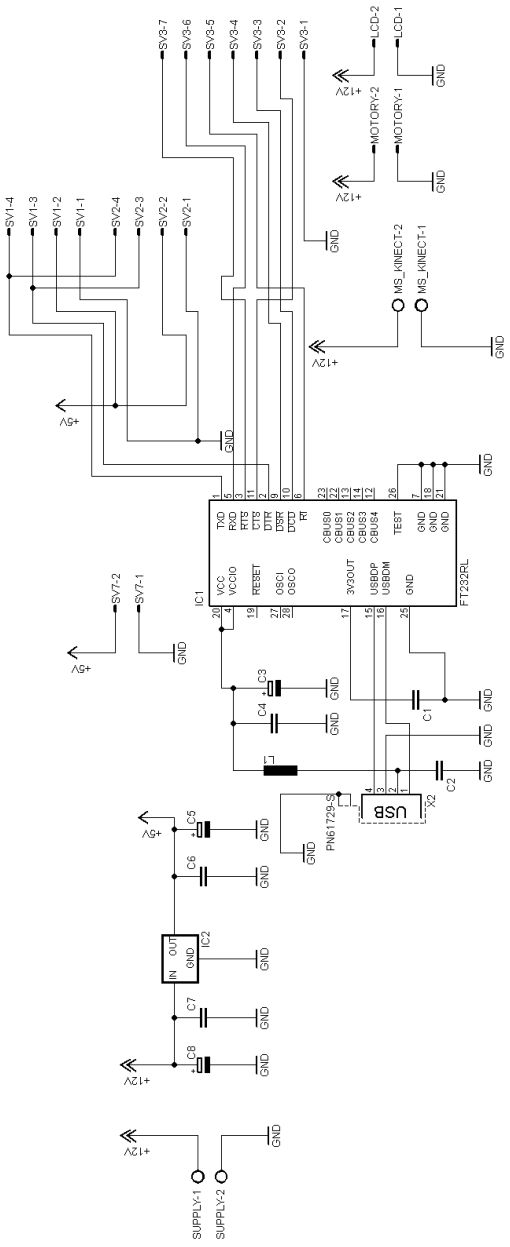
C SEZNAM POUŽITÝCH SOUČÁSTEK

Hodnota	**Název**	**Popis, hodnota**	**Počet**
IC1	FT232RL	Převodník USB/UART	1
L1	Civka	10uH	1
C1	Kondenzátor elektrolytický	100uF/25V	1
C2	Kondenzátor elektrolytický	10uF/25V	1
C3	Kondenzátor elektrolytický	10uF/25V	1
C4	Kondenzátor keramický SMD	100nF/50V	1
S	Svorkovnice	ARK-TL207B-2PGC	2
USB	US konektor do DPS	USB1X90B	1
SV1/SV2	Konektor - 4 piny	XINYA PFH02-04P	2
SV4/SV5/SV6	Konektor - 2 piny	XINYA PFH02-02P	3
SV1/SV2	Zásuvka - 4 piny	PSH02-04P	2
SV4/SV5/SV6	Zásuvka - 2 piny	PSH02-02P	3
-	Kontakt	XINYA PFF02-01FG	14
-	Krabička	U-KM35 ABS	1
-	Krabička	U-KP34	1
-	TFT monitor	7”TFT LCD monitor	1
Supply	Stabilizovaný zdroj	CP-S ABB 12V/10A	1
Konektor	HDE Power Supply Cable for Kinect		1

Tab. C.1: Seznam použitých součástek pro sériový modul a kompletaci robota.

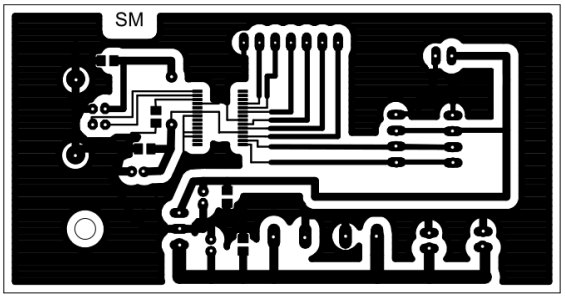
D SCHÉMA SÉRIOVÉHO MODULU A NÁVRH DPS

D.1 Schéma zapojení modulu



Obr. D.1: Schéma zapojení modulu.

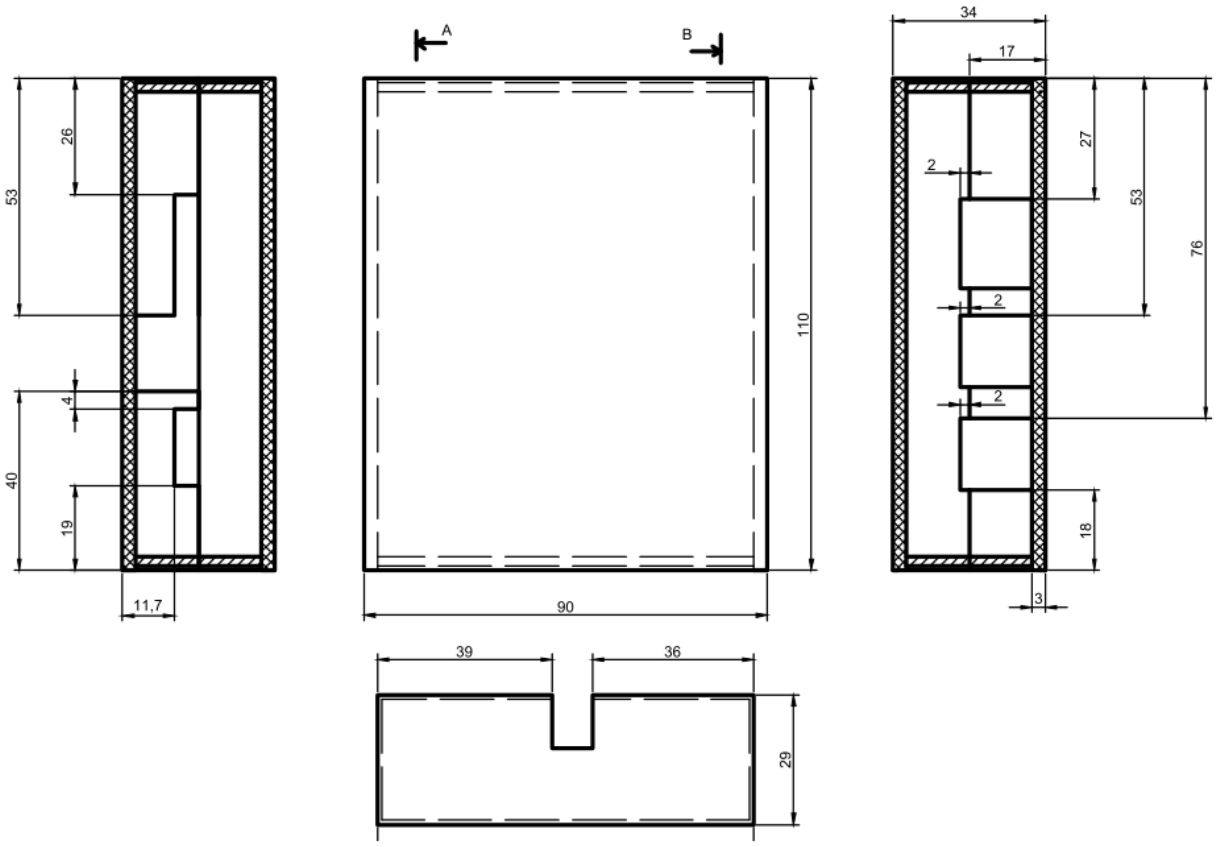
D.2 Navržené DPS



Obr. D.2: Navržená DPS Sériového modulu.

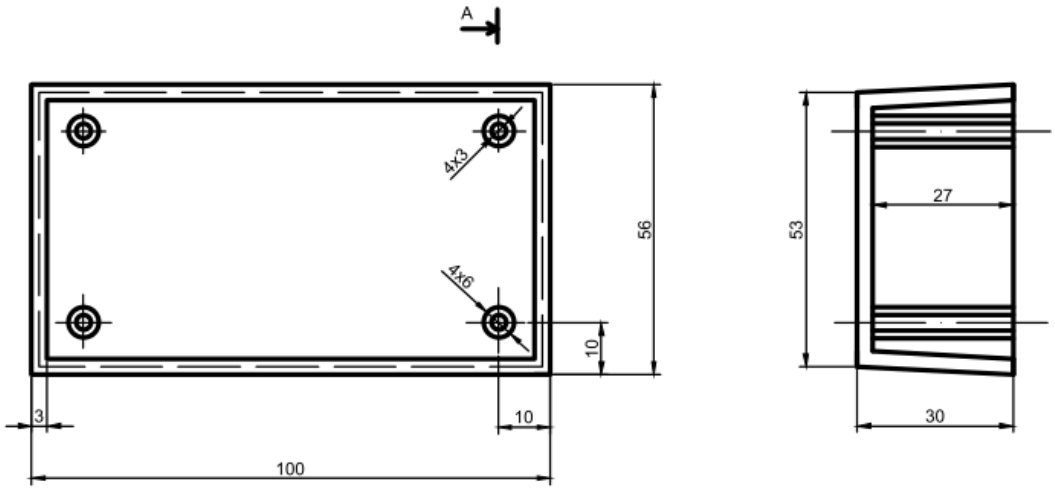
E MECHANICKÉ USPOŘÁDÁNÍ

E.1 Krabička pro umístění minipočítače



Obr. E.1: Parametry krabičky pro umístění BeagleBoardu-XM Rev.C.

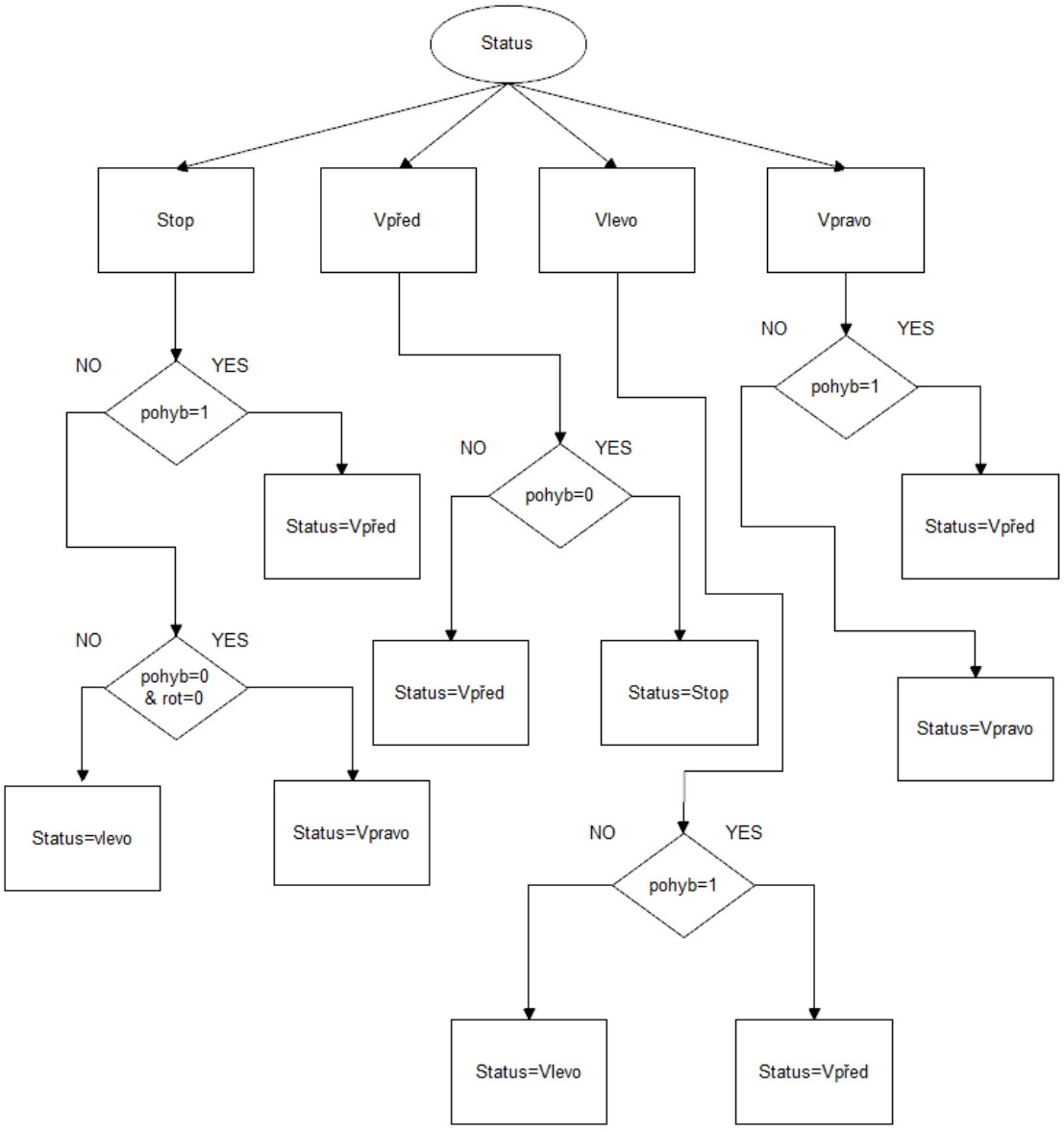
E.2 Krabička pro umístění DPS sériového modulu



Obr. E.2: Výkres krabičky pro umístění Sériového modulu.

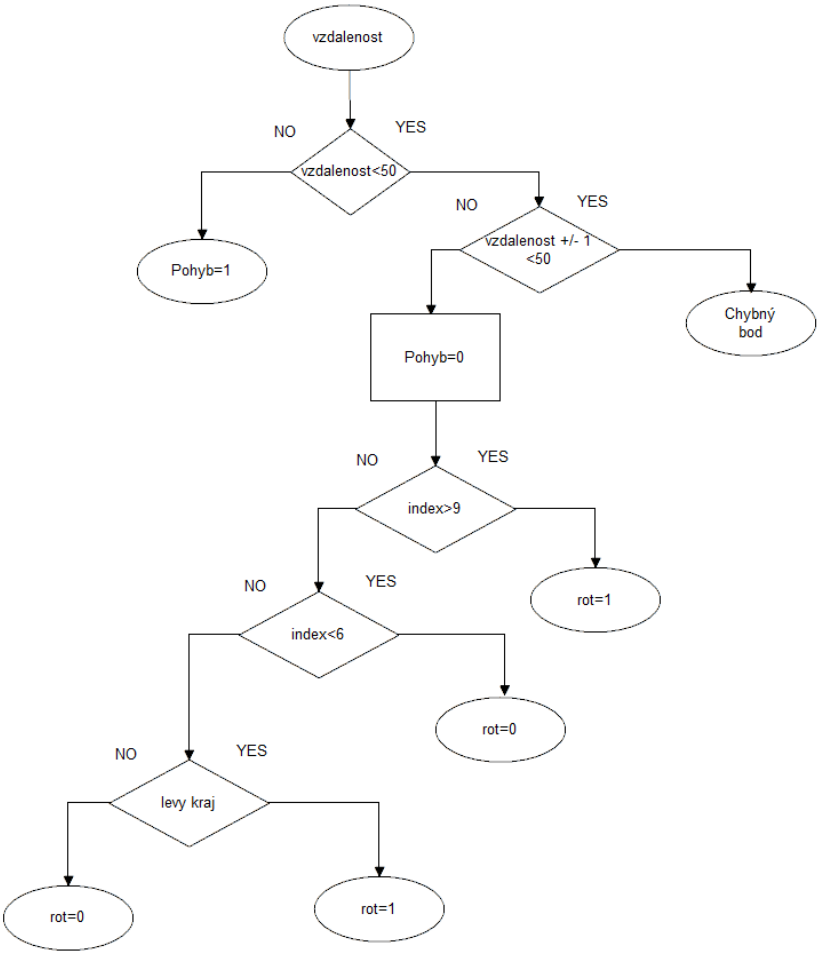
F VÝVOJOVÉ DIAGRAMY

F.1 Stavový automat



Obr. F.1: Stavový automat robota.

F.2 Vyhodnocení směru



Obr. F.2: Vývojový diagram pro zpracování a určení směru pohybu robota.

G VÝVOJÁŘSKÁ PŘÍRUČKA

G.1 Sériový modul

Nastavení výstupních signálů. Hodnoty SDA a SCL jsou relativní - záleží na hardwarové konfiguraci.

```
#define SDA 1
#define SCL 16
```

Definice adres jednotlivých slave zařízení:

```
#define SLAVE1 0x20
#define SLAVE2 0x22
```

Parametry FTDI obvodu:

```
#define VENDOR 0x0403
#define PRODUCT 0x6001
```

Základní struktura pro práci se Sériovým modulem. Tato struktura definuje veškeré možnosti modulu - jízdu vpřed, vzad, otáčení, nastavení rychlosti a zastavení.

```
typedef struct _sSerialModule
{
 ErrorMessage Acknowledgement;
 void (*Forward)();
 void (*Back)();
 void (*TurnLeft)();
 void (*TurnRight)();
 void (*Speed)(unsigned int rychlost);
 void (*Stop)();
}sSerialModule;
```

Inicializace struktury sSerialModule probíhá následující funkcí, které se předá pointer na referenci sSerialModule a ukazatel na funkci, která zpracovává stav modulu.

```
int SerialModuleInit(sSerialModule ** module, ErrorMessage message);
```

Zastavení a uvolnění paměti má na starosti následující funkce:

```
int SerialModuleStop(sSerialModule ** module);
```

G.2 MS Kinect

Klíčová struktura pro práci s Kinectem:

```
typedef struct _sKinect
{
 ErrorMessage Acknowledgement;
 void (*RealLenght)(float* data,int index);
 void (*GetVideo)(IplImage ** video);
 void (*GetDepth)(IplImage ** depth, short ** data); // Tresholded his.
 void (*Histogram)(IplImage ** histogram, short ** _data); // Normal his.
}sKinect;
```

Funkce, která inicializuje strukturu pro práci s Kinectem:

```
int KinectInit(sKinect ** kinect, ErrorMessage message);
```

Zastavení Kinectu a uvolnění alokované paměti má na starosti následující funkce:

```
int KinectStop(sKinect ** kinect);
```

G.3 Použité struktury a funkce OpenCV

Pro zpracování a zobrazení videa byly použity některé funkce a proměnné OpenCV.

G.3.1 CvPoint

Představuje bod zadaný pomocí souřadnic x a y.

```
typedef struct CvPoint
{
    int x;
    int y;
}
CvPoint;
```

G.3.2 IplImage

struktura pro zpracování a práci s obrázky.

```
typedef struct _IplImage
{
    int  nSize;
```

```
    int  ID;
    int  nChannels;
    int  alphaChannel;
    int  depth;
    char colorModel[4];
    char channelSeq[4];
    int  dataOrder;
    int  origin;
    int  align;
    int  width;
    int  height;
    struct _IplROI *roi;
    struct _IplImage *maskROI;
    void  *imageId;
    struct _IplTileInfo *tileInfo;
    int  imageSize;
    char *imageData;
    int  widthStep;
    int  BorderMode[4];
    int  BorderConst[4];
    char *imageDataOrigin;
}
IplImage;
```

H OBSAH PŘILOŽENÉHO CD

Přiložené CD obsahuje následující soubory:

- Elektronická verze diplomové práce,
- všechny zdrojové kódy programů uvedených v této práci,
- popis patentů firmy Prime Sense Ltd.,
- katalogový list obvodu FT232RL.