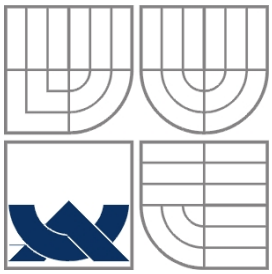


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

APLIKACE S 3D LASEROVÝM DÁLKOMĚREM SICK

3D LASER RANGE FINDER SICK APPLICATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Tomáš Fritz
AUTHOR

VEDOUCÍ PRÁCE — Ing. Tomáš Marada, Ph.D.
SUPERVISOR

BRNO 2009

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2008/2009

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Tomáš Fritz

který/která studuje v **magisterském navazujícím studijním programu**

obor: **Aplikovaná informatika a řízení (3902T001)**

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 diplomové práce:

Aplikace s 3D laserovým dálkoměrem SICK

v anglickém jazyce:

3D laser range finder SICK application

Stručná charakteristika problematiky úkolu:

Cílem této práce je implementovat sadu algoritmů pro zpracování 3D obrazu získaného z 2D laserového dálkoměru vertikálně polohovatelného.

Cíle diplomové práce:

1. Seznamte se s laserovým dálkoměrem SICK LMS 291.
2. Seznamte se s konstrukcí 3D laserového dálkoměru vertikálně polohovatelného.
3. Realizujte algoritmy pro čtení dat z dálkoměru a jejich následné zpracování.

Seznam odborné literatury:

[1] Firemní materiály týkající se laserového dálkoměru fy SICK.

Vedoucí diplomové práce: Ing. Tomáš Marada, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2008/2009.

V Brně, dne 12.11.2008

L.S.

doc. RNDr. Ing. Miloš Šeda, Ph.D. Ředitel ústavu	doc. RNDr. Miroslav Doupovec, CSc. Děkan fakulty

ABSTRAKT

Tato práce se zabývá vyžitím 3D laserového dálkoměru pro potřeby autonomních mobilních systémů. 3D skener je postaven jako rozšíření 2D laserového dálkoměru rotačním modulem.

V první části práce je popsán samotný laserový dalkoměr SICK LMS 291, jeho polohovatelná konstrukce a použité programové prostředky. Druhá část se zabývá návrhem a implemetací algoritmů pro čtení dat a jejich zpracování metodami rekonstrukce povrchu, oktalového stromu a segmentace objektů pomocí Houghovi transformace.

ABSTRACT

This thesis presents a use of 3D laser range finder designed for purposes of autnonomous mobile systems. The 3D scanner is built as extension of 2D laser range finder with rotation module.

In the first section is described laser range finder SICK LMS 291 and his pitching construction along with used software tools. Second part deals with design and implementation of algorithms for data reading and their processing with methods of surface reconstruction, octree and object segmentation with Hough transform.

KLÍČOVÁ SLOVA

SICK LMS 291, laserový dálkoměr, 3D mapování, oktalový strom, Houghova transformace

KEYWORDS

SIKC LMS 291, laser range finder, 3D mapping, octree, Hough transform

PODĚKOVÁNÍ

Tímto bych rád poděkoval panu Ing. Tomáši Maradovi, Ph.D. za všechny cenné rady, připomínky a odborné vedení, které mi poskytl během vzniku této práce.

Obsah:

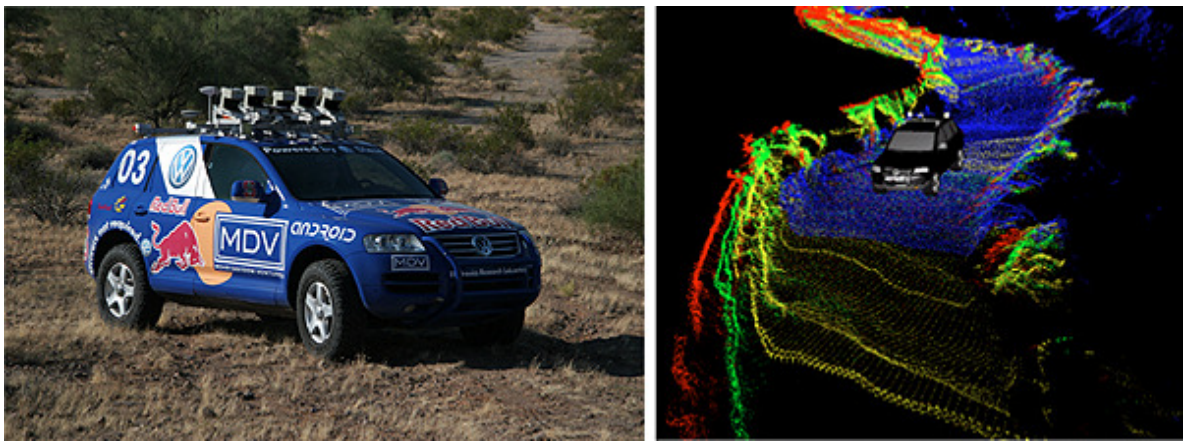
1 ÚVOD

Předpovědi na začátku devadesátých let tvrdily, že na sklonku tisíciletí bude v různých oblastech průmyslové výroby a servisních služeb nasazeno okolo 50 tisíc nezávisle operujících autonomních robotů. Skutečnost je však jiná. V průmyslovém prostředí jsou standardem stroje řízené magnetickou nebo optickou cestou.[1]

Autonomní mobilní systémy nebo také systémy neomezené cestou jsou stále používány velmi zřídka, ačkoliv se poptávka po nich stále zvyšuje. Jedním z několika důvodů propasti mezi předpověďmi a realitou byl nedostatek dobrých, levných senzorů. Situace se však v posledním desetiletí obrací k lepšímu a tak i velmi rychlé a přesné senzory se stávají stále dostupnějšími.

Ty jsou základním a jediným prvkem robotů, který jim umožňují vnímat okolní prostředí v reálném čase a konat na základě získaných dat. Obvykle se jedná o kamery, sonarové, laserové a infračervené dálkoměry, radary, taktilní senzory, kompasy a GPS. Jejich výstupní veličiny jsou následně transformovány na informace vhodné pro reprezentaci okolí robotu. Takováto zjednodušená reprezentace komplexního okolí se označuje jako mapa a celý proces jejího získávání jako mapování. Potřeba kvalitního mapování je obecně považována za jeden z nejdůležitějších problémů pro bezchybnou funkci ať už dálkově ovládaných, nebo plně autonomních mobilních robotů. Tato oblast navzdory značnému pokroku stále poskytuje velké výzvy.

Dobrým příkladem je dnes již velmi známá soutěž plně autonomních mobilních prostředků *DARPA Challenge* vypsaná americkou obranou organizací. Cílem druhého ročníku bylo projet trasu v pouštním prostředí v časovém limitu. Vítězem se stal vůz týmu vedeného Sebastienem Thrunem, který je jedním z průkopníků v oblasti navigace, lokalizace a mapování mobilních robotů a jedna z metod mapování publikovaná v jeho pracích [5] [8] bude popsána v dalších kapitolách.



Obr. 1.: Stanley, vítězný vůz DARPA Challenge a ukázka dat z jeho senzorů

Většina všech dnešních algoritmů mapování pracuje v 2D prostoru. 2D mapování se může zdát dostačující pro prostředí, které je statické, strukturované a limitováno velikostí. Mapování nestrukturovaného dynamického a rozsáhlého prostředí zůstává stále otevřenou oblastí vhodnou pro využití 3D modelování, které má oproti 2D důležité přednosti. Např. 3D mapy ulehčují rozpoznávání různých míst při lokalizaci robotu, které by se v 2D mohly zdát stejné, jelikož 3D model prostředí je bohatší než 2D model. Dalším nedostatkem pouze 2D informací mohou být problémy s přečnívajícími (konzolovitými)

objekty jako např. stoly, židle atp. 3D mapy jsou také více vhodné pro osoby zajímající se o interiéry budov, jako např. architekty nebo záchranné či vojenské složky, které se chtějí seznámit s nebezpečným prostředím, než do něj sami vstoupí.

Pro 3D mapování jsou používány stereoskopické kamery, sonary nebo laserové snímače. Komerční 2D laserové snímače jsou dostupné a v robotice dosti používané. Na druhé straně v podstatě neexistují komerční 3D skenery, které by byly vhodné pro použití u autonomních robotů v dynamickém prostředí. Laserové skenery používané v zeměměřičství nesplňují požadavky pro strojní automatizaci a robotiku, protože jsou dosti těžké, drahé a doby skenování jsou příliš dlouhé (až několik minut). Obvyklým řešením jak dosáhnout 3D skenování vhodného pro tyto aplikace je použití standardního 2D dálkoměru a mechanického pohonu k dosažení třetí dimenze.

V této práci je popsán přístup použitý řadou výzkumných skupin např. [1] [2]. Jedná se o využití 2D laserového dálkoměru rozšířeného o rotační modul poháněný servomotorem. Vylepšená konstrukce tak umožňuje 3D skenování, které překonává nedostatky informací pouze jedné roviny měření. Kombinace tohoto vylepšení spolu se sadou algoritmů pro získávání a zpracování dat ze skeneru poskytuje systém pro detekci překážek a 3D mapování prostředí.

2 POUŽITÉ HARDWAROVÉ PROSTŘEDKY

Laserové měření vzdálenosti se v posledních letech stalo důležitým prvkem v oblasti měřících metod. Laserové dálkoměry se začínají stále více používat také pro rychlou 3D digitalizaci okolního prostředí. Proces spočívá ve skenování fyzických objektů laserovým skenerem, který poskytuje informace o geometrii objektu a jeho umístění v prostoru a tak udává jeho 3D „otisk“. Díky této schopnosti se 3D skenery používají stále častěji jako čidla pro mapování prostředí a navigaci autonomních robotů.

2.1 SICK LMS 291 – Laserový měřící systém

Německá společnost SICK je jeden z hlavních výrobců technologií specializujících se na průmyslové senzory. Především pak na optická řešení v automatizaci, bezpečnostní systémy, automatické identifikace a měření.

SICK LMS 291 Laserový měřící systém je výkonným prvkem modelové řady LMS extrémně přesných a rychlých dálkoměrů určený pro průmyslové prostředí. LMS se stávají stále oblíbenějšími nejen v klasických aplikacích, ale i v oblastech autonomních robotů. Jedná se o bezkontaktní měřící systém, který skenuje svoje okolí ve dvou dimenzích (tzv. laserový radar - „LIDAR“). Jako některé jiné skenovací systémy ke své funkci nevyžaduje žádné speciální reflekční nebo poziční značky.

Obr. 2.: SICK LMS 291

Výstupem sériového rozhraní LMS je sada naměřených hodnot. V závislosti na dané aplikaci mohou být:

- Manuálně zobrazeny a vyhodnoceny na PC buďto pomocí dodávaného softwaru „LMSIBS“ (omezený objem dat), nebo pomocí vlastní aplikace, např. pro detekci polohy objektů a jejich velikostí.
- Získávány a vyhodnocovány v reálném čase na počítači pomocí dostatečně rychlé datové komunikaci.

Integrované vyhodnocovací rutiny také umožňují využití LMS přímo jako senzoru pomocí přepínaných výstupů v aplikacích monitorování oblasti (dvojdimenzionální). Pro

pokrytí větších oblastí mohou být použity dva synchronně operující LMS v master / slave konfiguraci.

LMS komunikuje s nadřazeným systémem pomocí přepínatelného RS 232 / 422 datového rozhraní. Specifikace LMS 291 nejdůležitějších parametrů je uvedena v tabulce.

SICK LMS 291	
Maximální dosah	80m
Zorné pole	100° / 180°
Úhlové rozlišení	0.25°/ 0.5°/ 1°
Doba odezvy	13 … 53 ms
Frekvence skenování	75 Hz
Rozlišení	10 mm
Systematická chyba	+/- 35 mm
Datové rozhraní	RS 232, RS 422
Přenosová rychlost	9,6 / 19,2 / 38,4 / 500 kBaud
Napajecí napětí	24 V DC +/- 15%
Příkon	20W
Hmotnost	4,5 kg
Rozměry	156 x 155 x 210 mm

Tab. 1.: Technická specifikace LMS 291 [12]

2.1.1 Princip laserového měření vzdálenosti

Základním principem zařízení aktivního bezkontaktního měření vzdálenosti je vyslání signálu (radiového, ultrazvukového nebo optického) na povrch objektu a zpracování odraženého signálu k určení vzdálenosti. Použití laseru v tomto kontextu vyniká nad ostatními typy signálu z několika následujících důvodů:

- Laser poskytuje koncentrovaný paprsek s velmi malým rozptylem a odchylkami. Radiové a ultrazvukové vlny nemohou takto malého rozptylu dosáhnout.
- Laserový paprsek má velkou intenzitu a díky malému rozptylu si tuto intenzitu zachovává na velké vzdálenosti.
- Monochromatičnost laserového paprsku obecně vede ke snazšímu zpracování signálu.

Laserový paprsek je obvykle produkován laserovou diodou a příslušnou optickou soustavou. Laserová dioda převádí elektrický signál na světelnou vlnu nebo impulz. Odražený signál je detekován pomocí lavinové fotodiody. Tato fotodioda převádí zpět světelnou vlnu nebo impulz na elektrický proud. Následně jsou použity vhodné členy pro zesílení, zpracovací a filtraci signálu. Jedním z nejčastějších matematických principů měření vzdálenosti je měření doby letu laserového pulzu.[11]

Princip této metody je jednoduchý. Laserový impulz je vyslán do okolí a následně je změřen čas, který paprsku zabere cestu k cíly, odražení a cestu zpět do detektoru. Pokud d je vzdálenost k cíly, t je doba odrazu a c je rychlost světla, potom [11]:

$$d = \frac{1}{2}ct \tag{2.1}$$

Pro bezchybné měření musí byt splněna podmínka délky impulzu, která musí být menší nežli doba odrazu t. Tedy

$$t > T_p \tag{2.2}$$

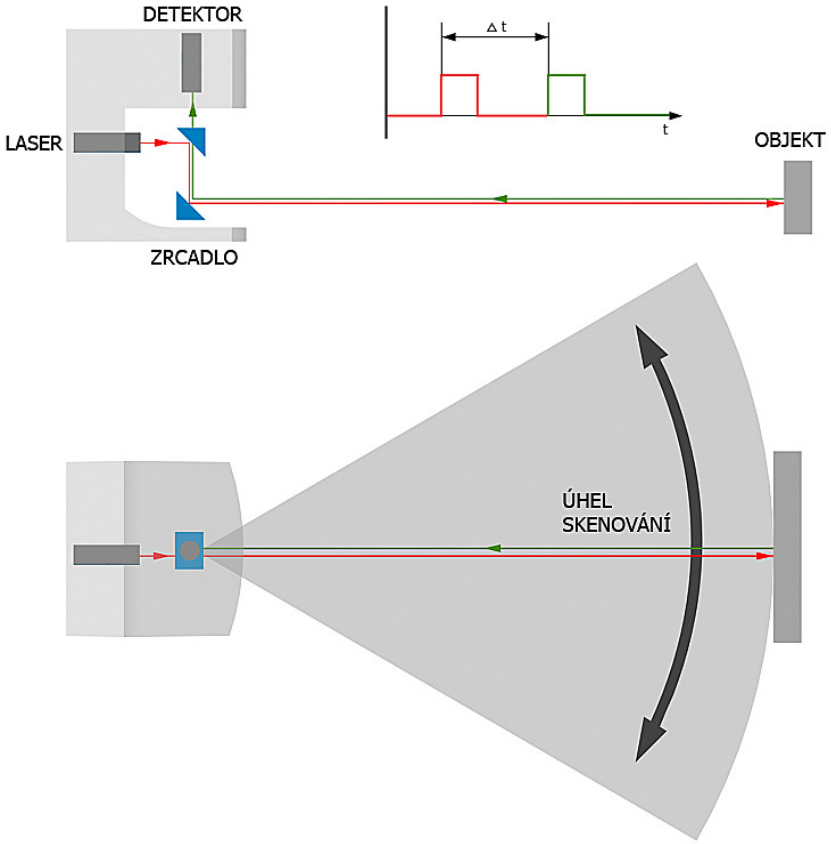
nebo

$$d > \frac{1}{2}cT_p \tag{2.3}$$

Z toho vyplývá, že hlavní problém spočívá v přesném změření doby odrazu t a tudíž chyba ve výpočtu je přímo úměrná t. Odchylka 1ps tedy dává chybu vzdálenosti $\delta d = 0.15$ mm.

$$\delta d = \frac{1}{2}c\delta t \tag{2.4}$$

Schéma 2D skeneru je zobrazeno na Obr. 3. Laserový paprsek znázorněný šipkami je vychýlen rotujícím zrcadlem. Tímto způsobem může 2D skener poskytnout body, které leží v půlkruhové rovině odklonu paprsku. Obrys cílového objektu je získán jako posloupnost přijímaných impulzů. 3D sken okolí je získán, pokud je rovina skenování otáčena v pravidelných intervalech okolo osy procházející těžištěm skeneru.



Obr. 3.: Princip laserového měření

Časová diskretizace je velmi důležitou součástí přesného měření času v těchto systémech. Úlohou diskretizace je sledování informace o času z elektrických impulzů detektoru a produkování spouštěcího signálu ve vhodných okamžicích. Volba derivační metody závisí na požadovaném dynamickém rozsahu a frekvenci pulzů. Běžně používané principy diskretizace zahrnují časování na náběhovou hranu impulzu (při konstantní amplitudě), „zero-crossing“ derivaci, integraci a poměrné časování. Princip spočívá v hledání okamžiku, kdy je velikost impulzu v konstantním poměru k amplitudě pulzu. Výskyt tohoto bodu znamená spuštění signálu.[11]

Díky fyzikálnímu principu bezkontaktních měřících zařízení je požadována jistá energie nositelského impulzu k překročení vnitřní spouštěcí prahové hodnoty. U měření doby letu energie odraženého světla závisí na vzdálenosti měřeného objektu od skeneru, ale také na povrchových charakteristikách objektu.[11]

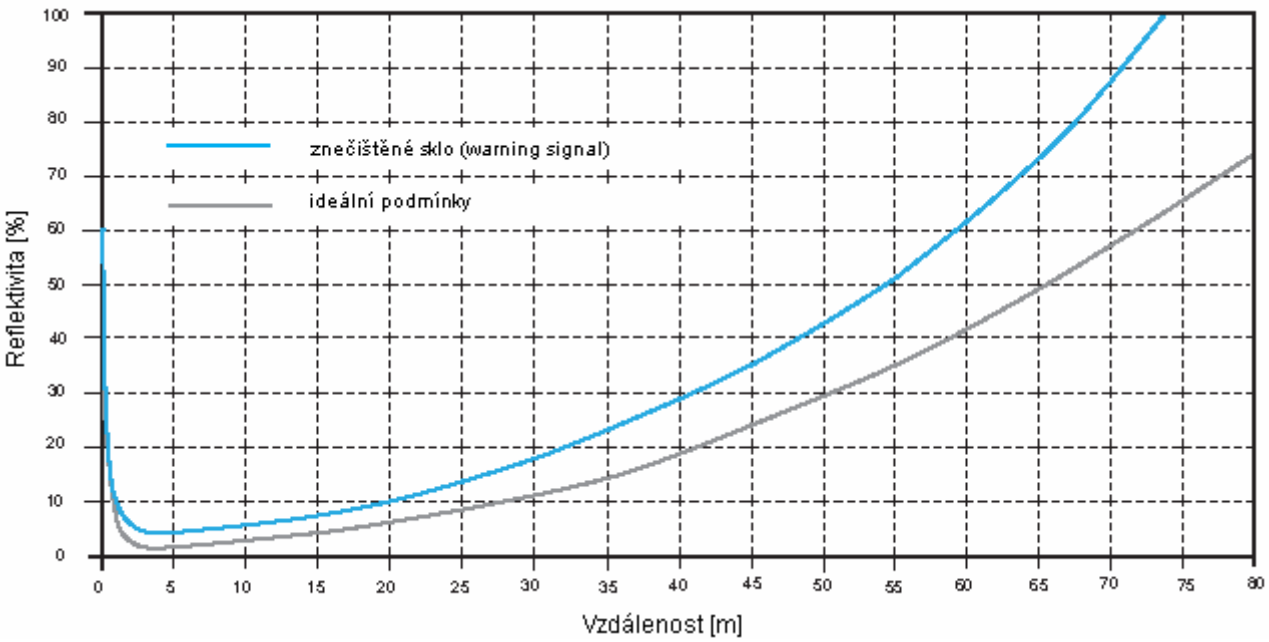
Dosah skeneru je závislí především na reflektivitě skenovaných objektů a na síle vysílaného paprsku. LMS používá přijatou energii k vyhodnocení vzdálenosti a porovnává ji s vnitřními referencemi. Hodnoty přijaté energie (reflektivity) jsou počítány pro vlnovou délku 905 nm. Reflektivita je vlastnost měřeného objektu, jeho schopnost objektu odrážet světlo, kterou nelze snadno zaznamenat. Naměřené hodnoty energetické úrovně z objektu se odvíjejí od reflektivity objektu, ale nejsou stejné jako jeho absolutní reflektivita. Některé hodnoty reflektivity pro obvyklé materiály jsou obsaženy v Tabulce 2. (standarty KODAK).[12]

Materiál	Reflektivita
Karton, černý	10%
Karton, šedý	20%
Dřevo	40%
PVC, šedé	50%
Papír, bílý	80%
Hliník	110% … 150%
Ocel	120% … 150%
Leštěná ocel	140% … 200%
Reflektor	> 2000%

Tab. 2.: Reflektivita základních materiálů [12]

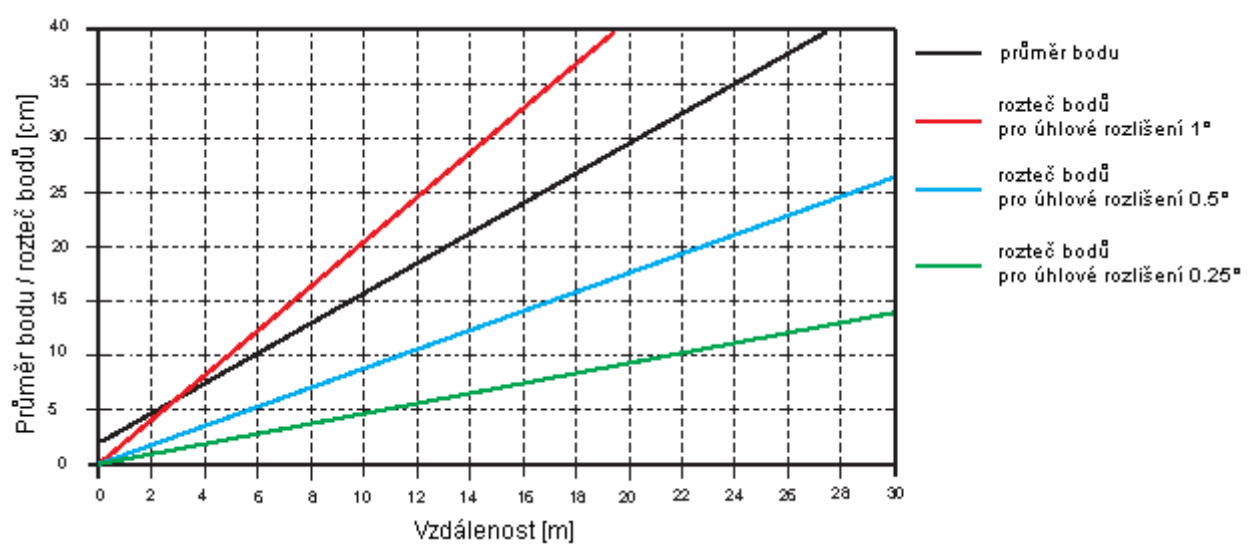
Tmavý testovací objekt v blízkosti skeneru může mít stejnou energetickou hodnotu jako světlý objekt ve větší vzdálenosti. Hodnoty reflektivity mohou pomoci popsat strukturní přechody materiálu ve stejné vzdálenosti. Lze ji tak využít např. pro zjišťování přechodu z bílé do černé ve stejné vzdálenosti měření.

Vztah mezi požadovanou reflektivitou materiálu a měřícím dosahem dálkoměru LMS 291 je znázorněn v Grafu 1., ze kterého je patrné, že za ideálních podmínek je dálkoměr schopen detekovat tmavé objekty s reflektivitou pouze 10% na vzdálenost až 30 metru. V rozmezí 0 – 1,5 metru je snížena citlivost, kvůli vlivu intenzity okolního světla na optickou soustavu, což snižuje v tomto rozmezí schopnost detekce tmavých objektů. V případě potřeby detekce tmavých, blízkých objektů se dají hodnoty citlivosti softwarově nastavit, to ovšem může způsobovat chyby měření při větší intenzivitě okolního osvětlení.



Graf 1.: Vztah mezi reflektivitou a dosahem dálkoměru [3]

Při vyhodnocování měření je nutno počítat s dvěma dalšími parametry. Emitovaný světelný paprsek má fyzický obvod / průměr, který se s rostoucí vzdáleností zvětšuje. Plocha paprsku při dopadu na objekt bývá označována jako bod. Jakožto bezkontaktní zařízení založené na měření světla potřebuje náležitou návratovou energii paprsku pro zastavení měření doby letu. Při dopadu paprsku na objekt ve větší vzdálenosti může nastat situace, kdy průměr paprsku je větší než šířka objektu, nebo jeho dopadu na hranu objektu. Odražené světlo pak nemusí mít dostatečnou energii pro sepnutí měření. S rostoucí vzdáleností se zvětšuje rozteč naměřených bodů, která taktéž může způsobovat chyby v detekci úzkých objektů. Poměr mezi vzdáleností a průměrem bodu spolu s roztečí bodů je vynesena v Grafu 2.



Graf 2.: Rozteč bodů úhlového rozlišení [3]

2.1.2 Měření a výstup dat

Laserové skenery řady LMS jsou optimalizovány pro měření vzdálenosti. Základní funkční princip systému je vyhodnocování prvního impulzu tzn., že první příchozí návratový signál spouští měření vzdálenosti a dodatečné návratové impulzy na cestě jsou ignorovány. Výhody tohoto principu jsou následující:

- Žádné rušení způsobené odlesky.
- Pokud je objekt detekován je jisté, že leží na cestě paprsku.

Kontinuální měření je jiný základní princip laserových skenerů, který znamená, že jeden měřící cyklus v rovině skenování proběhne vždy během jedné rotace zrcadla. Vzhledem k funkci zrcadla, jehož rotace nemůže být zastavena, musí být měření prováděno v poměrných úhlových hodnotách. Je zde předpoklad, že spolehlivost měření má přednost před datovou komunikací.

LMS skener potřebuje 13.32 ms pro jednu standardní rotaci, což koresponduje s měřící frekvencí 75 Hz. Měření je vždy prováděno po 1° krocích, tzn., že pokud je nastaven krok 0.5° nebo 0.25° je zapotřebí dvou nebo čtyř rotací zrcadla.

K zajištění příslušného úhlového rozlišení jsou 1° kroky posunuty o příslušnou hodnotu (0.5°, 0.25°) na začátku otáčení zrcadla. To znamená, že sken s rozlišením 0.5° zabere 26.64 ms a krok 0.25° vyžaduje 53.28 ms. Výstup naměřených hodnot je řazen vzestupně podle měřených úhlů.

2.1.3 Formát dat a přenosové rychlosti

Formát dat přenášených pomocí rozhraní RS 232 / 422 je následující:

- 1 start bit
- 8 data bitů
- 1 stop bit

To znamená, že každý přenesený data byte je 10 bitů dlouhý. Dostupné přenosové rychlosti jsou popsány v Tab. 3. Po spuštění LMS je vždy přenosová rychlost nastavena na 9 600 Bd. Pro nejmenší úhlové rozlišení 0.25° a rozsah skenovaní 180° je nutné zajistit přenosovou rychlost 500 kBd jinak může docházet ke ztrátám informací při přenosu. Tuto rychlost podporuje pouze rozhraní RS 422.

Přenosová rychlost dat	Rozhraní	Poznámka
9 600 Bd	RS 232 / 422	Standardně po spuštění
19 200 Bd	RS 232 / 422	
38 400 Bd	RS 232 / 422	
500 kBd	RS 422	Pouze pro RS 422

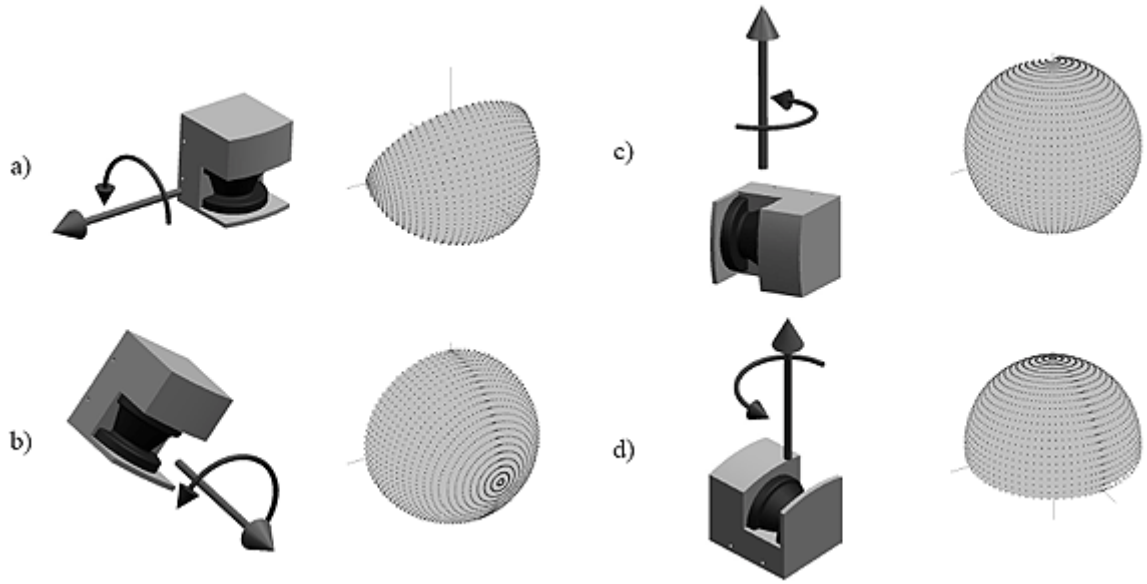
Tab. 3.: Přenosové rychlosti SICK LMS 291

2.2 Konstrukce 3D skeneru

Jelikož v podstatě neexistují vhodné komerční 3D skenery je pro 3D skenování běžnou praxí použití klasický 2D skeneru a nějakého mechanického systému rotace. V podstatě existují dvě metody. První metodou používanou u autonomních robotů je pevné upevnění skeneru na tělo robotu. 3D skenu je poté docíleno otáčením samotného robotu. Tato metoda má svá omezení např. robot není schopen detekovat pohybující se předměty a celkový proces skenování je náročnější. Častější metodou je připevnění 2D skeneru na rotační mechanismus ovládaný servomotorem.

Kombinace 2D laserového dálkoměru a servomotoru umožňuje různé uspořádání rovin skenování, os rotace a zorného pole. Každé uspořádání má své specifické výhody a nevýhody. Obsáhlý přehled je uveden v [10]. Vzhledem k nepříliš vypovídajícím překladům pojmenování různých metod budu dále uvádět anglická označení.

Prvním velmi častým způsobem je otáčení skeneru kolem horizontální osy (tzv. „pitching scan") jak je zobrazeno na Obr. 4a. Zde je rovina skenování horizontální s možným natočením v rozsahu 180°. Druhým často používaným uspořádáním (Obr. 4c.) je rotace kolem vertikální osy, kdy je skener natočen o 90° (tzv. „yawing scan"). U tohoto uspořádání je možný rozsah skenování až 360° bez nutnosti pohybu robotu při vhodně navrženém rotačním mechanismu. Nově používanou metodou se začíná stávat tzv. rolling scan (Obr. 4b.). Jeho předností je především největší hustota bodů přímo před skenerem, která může být v řadě případů velmi žádoucí (např. detekce osob).



Obr. 4.: Možnosti rotace skeneru: a) pitching scan b) rolling scan c) yawing scan d) yawing scan top [10]

Rozdíl mezi nejčastěji používanými přístupy *pitching scan* a *yawing scan* je v orientaci vrcholového úhlu a vlivu pohybujících se objektů na získaná data. U druhého způsobu uspořádání může s jistou pravděpodobností nastat problém s pohybujícími se objekty. Pokud by se objekt pohyboval stejnou rychlostí, jakou je prováděno skenování, tak by mohly být naměřená data zcela znehodnocena. Stejná situace by v prvním případě sice taktéž ovlivnila naměřená data, ale ne tak zásadně, aby nemohla být dále zpracována.

V této práci je skener namontován na vertikální stojan a pomocí servomotoru je

s ním otáčeno kolem horizontální osy (*pitching scan*). Realizace polohovacího mechanismu byla předmětem bakalářské práce [21].

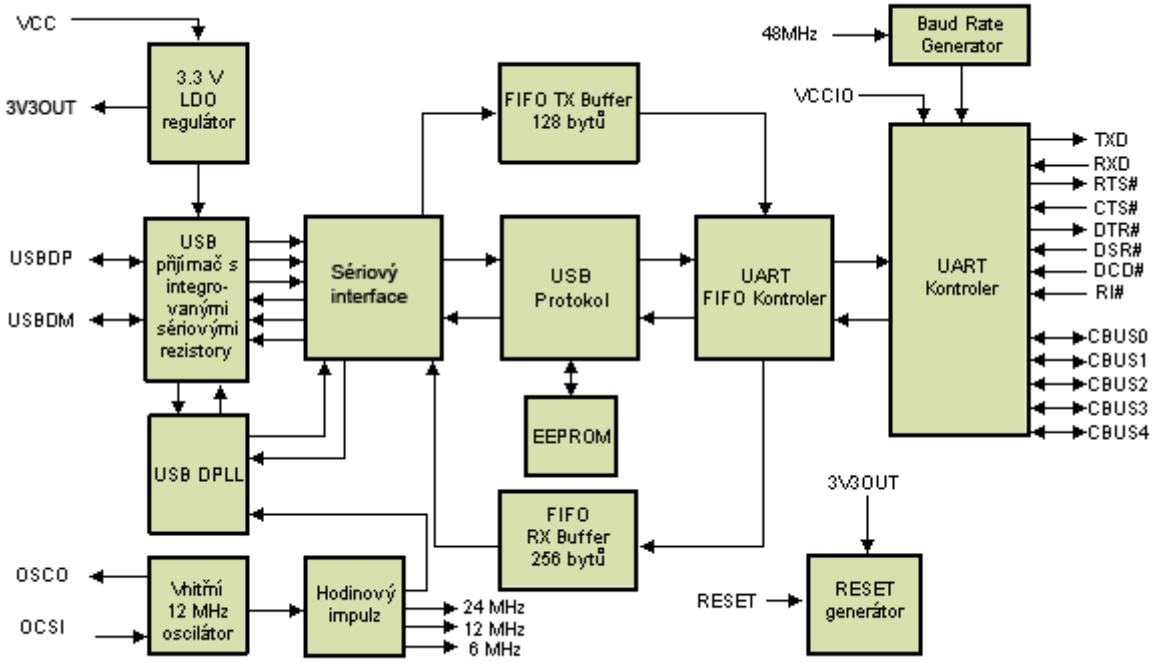
Obr. 5.: Konstrukce 3D skeneru

2.3 Obvod FTDI 232RL

Jak již bylo zmíněno výše LMS SICK standardně poskytuje přenosové rychlosti pouze do 38 400 Baudu pro rozhraní RS-232. Pro dosažení rychlosti 500 kBaudu pomocí RS-422 výrobce nabízí dosti drahou MOXA kartu. Mnohem levnějším, ale ne horším řešením je použití běžně dostupného a levného obvodu FT232RL společnosti FTDI.

Obvod FT232RL tedy umožňuje převod standardní komunikační linky USB na rozhraní RS-422. Díky tomu uživatel nepotřebuje podrobně znát detailní způsob komunikace přes USB sběrnici a programovat pro ni obslužné algoritmy ve své aplikaci. Hlavní vlastnosti obvodu lze shrnout v těchto bodech [16]:

- Jednočipové rozhraní pro převod USB na asynchronní sériová data
- Přenosové rychlosti 300Bd až 3MBd (RS232, RS422, 485) v TTL úrovních
- 256 byte RX buffer a 128 byte TX buffer
- FIFO pro příjem i vysílání
- Podpora řízení přenosu HW i SW
- Podpora ve všech běžných OS
- Programovatelné polarity signálu, pomocné signály
- FTDI VCP a D2XX ovladače pro snadnou obsluhu
- Nízká cena



Obr. 6.: FT232RL Blokový diagram [16]

3 POUŽITÉ SOFTWAROVÉ PROSTŘEDKY

3.1 Platforma .NET

Platforma .NET byla oficiálně představena firmou Microsoft v roce 2000 jako klíčový produkt, jehož rozvoj a propagace je součástí dlouhodobé strategie firmy. Microsoft .NET znamená novou generaci systému vývoje aplikací pro operační systémy Windows založeném na řízeném běhovém prostředí, obohaceném o neskromnou sadu základních tříd, nesoucím jméno .NET framework. Hlavními důvody vedoucí k více než čtyřletému vývoji, jehož výsledkem je .NET, byly [19]:

- nekompatibilita jednotlivých programovacích jazyků a s tím související obtížná spolupráce mezi programy / knihovnami napsanými v odlišných jazycích (např. C++ a Visual Basic)
- vysoká chybovost aplikací (chyby v práci s pamětí, neplatné konverze datových typů)
- problémy s verzemi knihoven (obtížná práce s provozem více verzí knihoven)
- zastaralý a nepřehledný způsob vývoje dosavadních webových aplikací

Všechny tyto problémy efektivně řeší platforma .NET – a to použitím již zmíněného řízeného běhového prostředí, systémem assemblies, což jsou základní stavební prvky aplikací, a novou technologií ASP .NET pro vývoj webových aplikací.[19]

3.1.1 Princip běhového prostředí

Většina dnešních aplikací, vytvořených například v jazyce C++, Visual Basicu nebo Delphi, jsou zkompilovány přímo pro danou platformu, nejčastěji je to pro platformu Win32 operačních systémů Windows, ale mohou to samozřejmě být i jiné. To znamená, že zdrojový kód je kompilací převeden do strojového kódu počítače. To ve výsledku přináší velmi dobrou rychlost běhu výsledné aplikace. Avšak na druhou stranu z toho plynou i některé nevýhody – nepřenositelnost mezi jednotlivými platformami, popřípadě verzemi operačních systémů a nezřídka jsou k vidění chyby v přístupech do operační paměti.

Princip řízených běhových prostředí, použitý právě u platformy .NET, ale i u velmi známé platformy Java firmy Sun Microsystems, přidává k převodu zdrojového kódu do kódu strojového ještě jednu vrstvu. Tuto vrstvu představuje mezikód, do kterého jsou zdrojové kódy zkompilovány, a tento mezikód je běhovým prostředím na cílové platformě (Windows, Linux) převeden do strojového kódu. Tento převod je na cílové platformě realizován vždy při spouštění konkrétní aplikace. [19]

3.1.2 Klíčové vlastnosti

Mezikód zmíněný o pár řádků výše se ve světě této platformy nazývá MSIL, tedy Microsoft Intermediate Language. Tento jazyk relativních adres je spouštěn klíčovou součástí .NET frameworku pojmenovanou CLR (Common Language Runtime neboli společné běhové prostředí).

V prostředí CLR existuje věc, která programátorům velmi usnadňuje práci

s operační pamětí – Garbage Collector. Jedná se o sadu složitých algoritmů pro uvolňování nepotřebných programových objektů z paměti. Díky Garbage Collectoru se již vývojáři nemusí starat o přiřazování nebo uvolňování operační paměti a odpadá tak riziko již zmíněné nekorektní práce s ní, která ve většině situací končí pádem aplikace. [19]

Pro vývoj .NET aplikací je možné použít jeden z několika programovacích jazyků vyšší úrovně. Může se jednat například o:

- C#
- Visual Basic .NET
- J#
- managed C++

3.2 Jazyk Visual C# 2.0

C# (čteno jako anglické "C sharp") je jednoduchý, moderní, silně objektově orientovaný programovací jazyk odvozený z jazyka C a C++. Spojuje vysokou produktivitu Visual Basicu a hrubou sílu jazyka C++. Microsoft přichází s návrhem tohoto nového programovacího jazyka po definitivním rozchodu s Javou firmy Sun Microsystems.

V tomto jazyce je realizováno 80% základních knihoven .NET frameworku. I přesto, že je koncipován hlavně pro psaní řízeného kódu, na jehož užití je platforma .NET postavena, lze jej v případě potřeby využít i pro tvorbu kódu neřízeného(bloky unsafe). Použití neřízeného kódu znamená, že běhové prostředí CLR neověřuje zda-li je napsaný kód bezpečný (například se neověřuje jinak vyžadovaná typová bezpečnost). Zdrojový kód pro program psaný v C# je uchováván v jednom nebo více textových souborech s příponou `.cs`; při přeložení je vytvořen spustitelný `.exe` soubor.

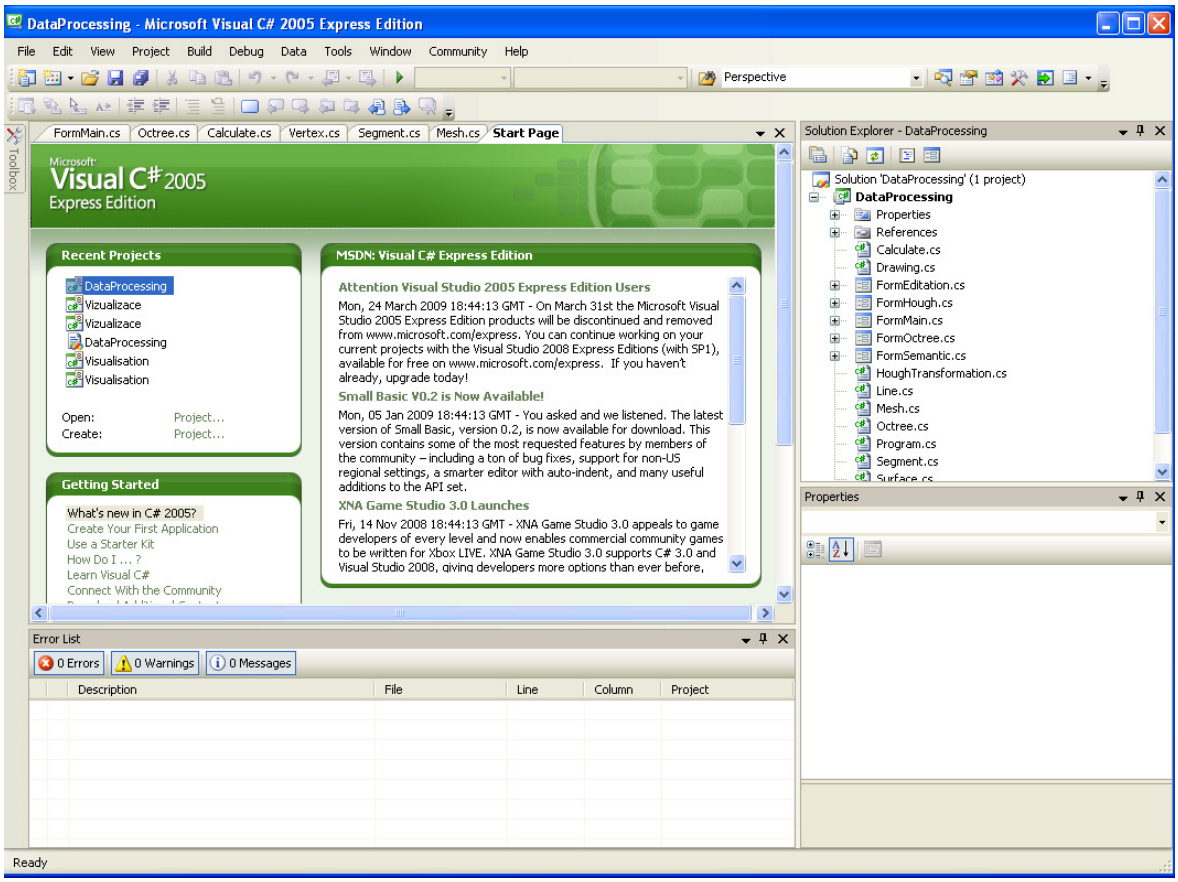
S příchodem C# 2.0 se na scénu dostává několik důležitých nových rysů, včetně generických typů a metod, iterátorů a anonymních metod. Ve vývojovém prostředí Microsoft Visual Studio 2005 se tyto výkonné novinky dají velmi snadno používat, produktivitu práce vývojáře pak výrazně zvyšují noví průvodci a různá další vylepšení, která jsou součástí Visual Studia 2005. Mezi základní vlastnosti jazyka C#, které lze při tvorbě aplikací použít patří [19]:

- Třídy – základní stavební prvek při tvorbě objektově orientovaných aplikací obsahující akce (metody) a atributy
- Struktury – lze je chápat jako zjednodušené třídy, jejich užitím jsou nejčastěji popisovány vlastní datové struktury.
- Výčtové typy
- Vlastnosti – někdy označované jako chytré proměnné
- Pole a jejich „chytrá" verze nazývaná indexery
- Delegáti – typově bezpečné ukazatele na funkce
- Události – druh delegátů sloužící ke zpracování asynchronních operací

3.2.1 Vývojové prostředí Visual C# 2005 Express

Visual C# 2005 Express je základní verzí MS Visual Studio 2005, která je určená především pro studenty. Visual Studio je vývojové prostředí (IDE), které může být použito

pro vývoj konzolových aplikací a aplikací s grafickým rozhraním spolu s Windows Forms aplikacemi a webovými aplikacemi. Visual C# 2005 Epress obsahuje řadu nástrojů jako je editor kódu podporující IntelliSense a refaktorování, designéra formulářů Windows pro tvorbu GUI aplikací, integrovaný grafický debugger apod.



Obr. 7.: Microsoft Visual C# 2005 Express Edition

3.3 OpenGL

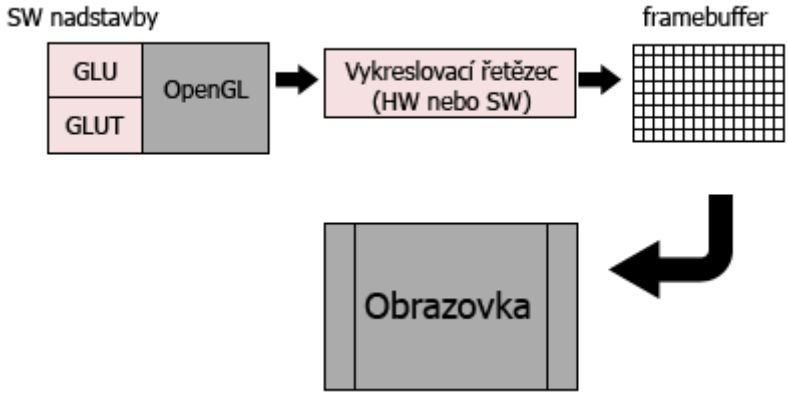
Knihovna OpenGL (Open Graphics Library) byla navržena firmou SGI (Silicon Graphics Inc.) jako aplikační programové rozhraní (Application Programming Interface - API) k akcelerovaným grafickým kartám resp. celým grafickým subsystémům. Předchůdcem této knihovny byla programová knihovna IRIS GL (Silicon Graphics IRIS Graphics Library). OpenGL byla navržena s důrazem na to, aby byla použitelná na různých typech grafických akcelerátorů a aby ji bylo možno použít i v případě, že na určité platformě žádný grafický akcelerátor není nainstalován - v tom případě se použije softwarová simulace. V současné době lze knihovnu OpenGL použít na různých verzích unixových systémů (včetně Linuxu a samozřejmě IRIXu), OS/2 a na platformách Microsoft Windows.[20]

Na některých platformách je možné rozdělení aplikace na dvě relativně samostatné části - serverovou a klientskou. Při vykreslování se potom jednotlivé příkazy (což jsou většinou parametry funkcí OpenGL) přenášejí přes síťové rozhraní. Knihovna OpenGL (narozdíl od IRIS GL nebo Direct 3D) byla vytvořena tak, aby byla nezávislá na použitém operačním systému, grafických ovladačích a správcích oken (Window Managers). Proto

také neobsahuje žádné funkce pro práci s okny (otevírání, zrušení, změnu velikosti), pro vytváření grafického uživatelského rozhraní (Graphical User Interface - GUI) ani pro zpracování událostí. Tyto funkce lze zajistit buď přímo voláním funkcí příslušného správce oken, nebo lze použít některou z nadstaveb, například knihovnu GLUT (OpenGL Utility Toolkit). Pro dosažení co největší nezávislosti na použité platformě zavádí knihovna OpenGL vlastní primitivní datové typy, například GLbyte, GLint nebo GLdouble. [6]

Programátorské rozhraní knihovny OpenGL je vytvořeno tak, aby knihovna byla použitelná v téměř libovolném programovacím jazyce. Primárně je k dispozici hlavičkový soubor pro jazyky C a C++. V tomto souboru jsou deklarovány nové datové typy používané knihovnou, některé symbolické konstanty (např. GL_POINTS) a sada cca 120 funkcí tvořících vlastní rozhraní.[20]

Z programátorského hlediska se OpenGL chová jako stavový automat. To znamená, že během zadávání příkazů pro vykreslování lze průběžně měnit vlastnosti vykreslovaných primitiv (barva, průhlednost) nebo celé scény (volba způsobu vykreslování, transformace) a toto nastavení zůstane zachováno do té doby, než ho explicitně změníme. Výhoda tohoto přístupu spočívá především v tom, že funkce pro vykreslování mají menší počet parametrů a že jedním příkazem lze globálně změnit způsob vykreslení celé scény, například volbu drátového zobrazení modelu (wireframe model) nebo zobrazení pomocí vyplněných polygonů (filled model). Vykreslování scény se provádí procedurálně - voláním funkcí OpenGL se vykreslí výsledný rastrový obraz uložený v tzv. framebufferu, kde je každému pixelu přiřazena barva, hloubka, alfa složka popř. i další atributy. Z framebufferu lze získat pouze barevnou informaci a tu je možné následně zobrazit na obrazovce.



Obr. 8.: Princip vykreslování grafiky pomocí OpenGL

Pomocí funkcí poskytovaných knihovnou OpenGL lze vykreslovat obrazce a tělesa složená ze základních geometrických prvků, které se nazývají grafická primitiva. Mezi tato primitiva patří bod, úsečka, trojúhelník, čtyřúhelník atd. Existují i funkce, které podporují proudové vykreslování některých primitiv - lze například vykreslit polyčáru (line loop), pruh trojúhelníků (triangle strip), pruh čtyřúhelníků (quad strip) nebo trs trojúhelníků (triangle fan). Na jednotlivá grafická primitiva lze aplikovat různé transformace (otočení, změna měřítka, posun, perspektivní projekce).[20]

Hlavičkový soubor pro jazyk C# primárně k dispozici není, proto je nutné použití některé z dostupných knihoven. Pro tuto práci byla zvolena knihovna Open Toolkit.

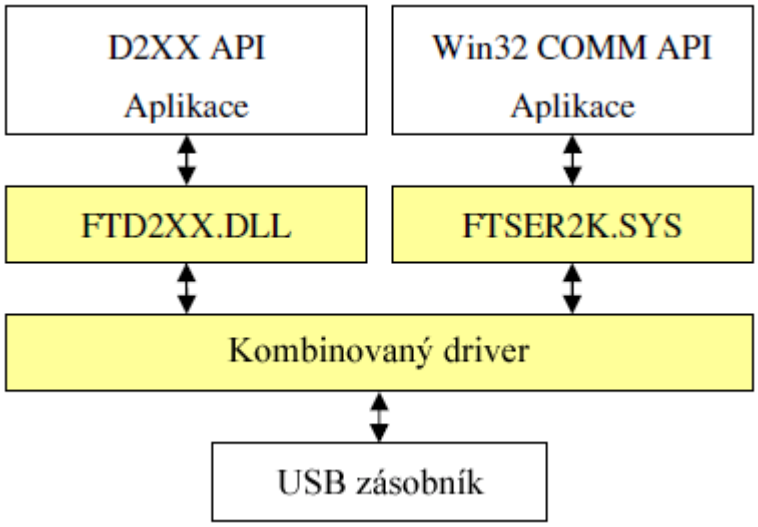
Open Toolkit je knihovna volně distribuovaná pod MIT/X11 licencí, která umožňuje použití OpenGL, OpenAL a OpenCL na platformě .NET a všech jejich jazyků. Jedná se o bohatou a typově bezpečnou sadu OpenGL funkcí. Obsahuje také matematické funkce pro práci s vektory maticemi.

3.4 Knihovny a drivery FTDI

Nová generace driveru pro obvody FTDI používá koncepci kombinovaného driveru (na rozdíl od dřívějších verzí) a stačí tak jediná sada driveru, která dává k dispozici jak rozhraní pro virtuální COM port, tak i proprietární rozhraní FTDI. Rozhraní Virtual Com Port umožňuje aplikacím přes Win32 COM API komunikovat s převodníkem jako se standardním sériovým portem. Zpřístupnění tohoto rozhraní je možné zakázat v konfiguraci součástky (v paměti EEPROM) nebo v konfiguraci driveru (v příslušném INI souboru).[17]

Pro komunikaci je možno použít libovolný terminálový program a vybrat si nově vzniklý COM port. Na rozdíl od obyčejného COM portu dojde k přerušení spojení mezi programem (terminálem) a USB COM portem kdykoli odpojíme a připojíme USB zařízení. Spojení je pak nutné znovu navázat. [17]

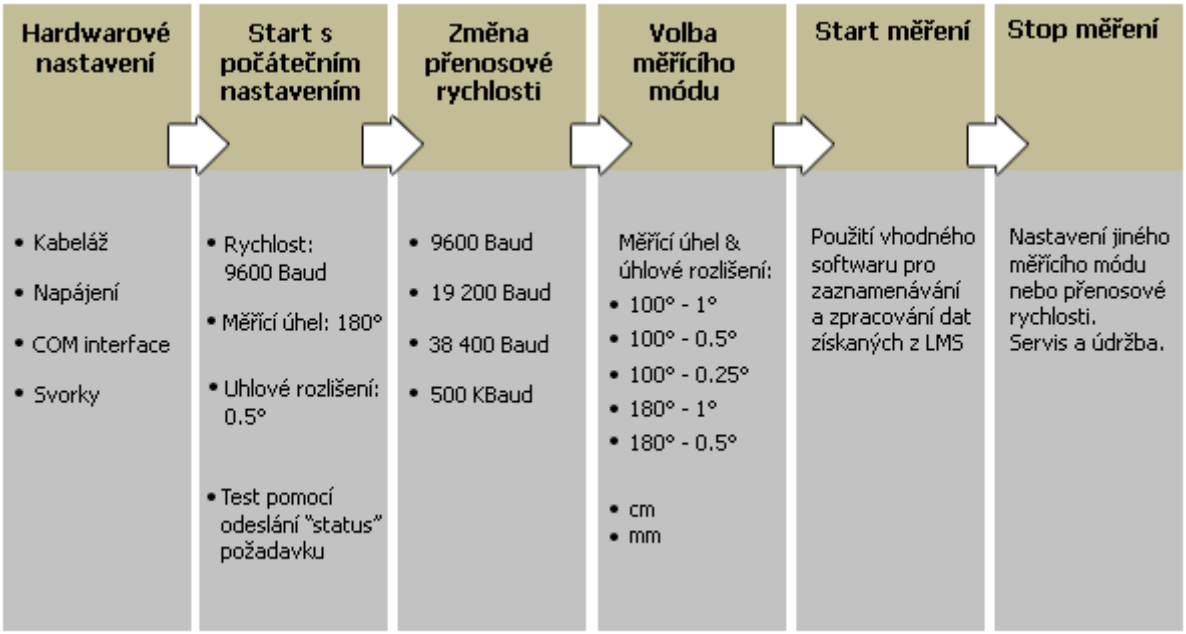
Druhou a v této práci použitou alternativou k VCP je rozhraní D2XX API, které umožňuje komunikaci s obvody FTDI včetně ovládání jejich speciálních funkcí pomocí DLL knihovny. Architektura D2XX obsahuje Windows WDM driver, který komunikuje s obvody skrze Windows USB zásobník a DLL, které poskytuje rozhraní pro aplikační software. Ze základních funkcí lze zmínit např. funkce pro načtení připojených jednotek, otevření jednotky, čtení a zápis dat nebo nastavení přenosové rychlosti.[15]



Obr. 9.: Architektura FTDI driveru

4 NAVRŽENÉ ŘEŠENÍ PRO ČTENÍ DAT

Prvním krokem pro využití LMS bylo vytvoření aplikace pro komunikaci se skenerem a získávání dat. LMS je nastavováno a ovládáno v krocích pomocí různých příkazů - telegramů. Od spuštění skeneru až po provedení měření musí proběhnout mezi LMS a řídicím systémem (PC) několik po sobě jdoucích procesů. Nejprve, po spuštění skeneru, je nutné navázat komunikaci přes USB převodník a získat počáteční nastavení skeneru pomocí tzv. *status* požadavku. Pokud je komunikace bez problému navázaná a bylo získáno počáteční nastavení skeneru, lze pokračovat dalším nastavováním. Lze změnit přenosové rychlosti, měřící úhly a jednotky měřených dat. Po nastavení vhodných parametrů, lze již spustit měření ve vhodném vertikálním úhlovém rozsahu, který je umožněn polohovatelnou konstrukcí skeneru. Celý postup tvořený z jednotlivých kroků je znázorněn na obrázku a podrobněji popsán dále.



Obr. 10.: Jednotlivé kroky ovládání skeneru

4.1 Struktura komunikačních telegramů

Lze rozlišit dva typy telegramů. První jsou telegramy odeslané z řídicího systému do LMS a dále budou označovány jako *telegramy příkazů*. Druhým typem jsou telegramy z LMS přijaté řídicím systémem, které budu označovat jako *telegramy odpovědí*. Každý telegram příkazu obsahuje vždy pouze jeden řídící *příkaz* pro LMS, stejně tak telegram odpovědi obsahuje vždy pouze jednu *odpověď* z LMS. Každý telegram příkazu je vždy následován telegramem odpovědi. *Příkazy a odpovědi* jsou vždy pouze jeden byte dlouhé.

Za *příkazem / odpovědí* obvykle následuje různý počet *data* bytu. Celý telegram má specifický rámec ohraničující příkazy a data. Pro odeslání dat do LMS musí mít telegram strukturu uvedenou v Tab. 4. LMS odpovídá stejnou strukturou.

Délka celého telegramu příkazu odeslaného do LMS a následujícího telegramu odpovědi může být různá v závislosti na konkrétním příkazu nebo měřícím módu. Například při nastaveném úhlovém rozlišení 0.5° a rozsahu skenování 180° dojde při

měření k zaznamenání 361 hodnot. Každá naměřená hodnota je reprezentována dvěma byty tedy celková délka datové části telegramu bude 722 bytů. Telegram nekončí znakem ETX (End of Text), ale kontrolním součtem (CRC). V telegramech odpovědí vždy CRC předchází status byte, který je pevnou částí data bloku.[13]

Označení části telegramu		Délka dat v bitech / délka dat v bytech / typ dat	Popis
STX (Start of Text)		8 / 1 / BYTE	Start byte (02h).
Adresa		8 / 1 / BYTE	Adresa odesilatele. Adresa může být použita např. pro odlišení více měřících zařízení.
Délka		16 / 2 / WORD	Počet následujících data bytů mimo kontrolní součet.
Příkaz / odpověď		8 / 1 / BYTE	Příkaz / odpověď
Data	Data odeslaného telegramu	N x 8 (n x 1)	V telegramech příkazů to mohou být rozšíření příkazu a / nebo limitní hodnoty.
	Status (pouze v telegramu odpovědi)	8 / 1 / BYTE	Status komunikace. Řídící systém nesmí ve svých telegramech odesílat status.
Kontrolní součet		16 / 2 / WORD	CRC kontrolní součet pro celý telegram počínaje STX po status byte.

Tab. 4.: Struktura telegramu

Všechny telegramy odeslané do LMS mají za následek dvě po sobě jdoucí odezvy. Pokud má telegram přijatý v LMS správnou strukturu, tak je zpracován a nazpět je odesláno potvrzení *Acknowledge* (ACK) (06h). Poté je odeslán korespondující telegram odpovědi. Pokud je v LMS přijatý telegram nesprávný je odesláno pouze nepotvrzení *Not Acknowledge* (NAK) (15h). Ukázkový příklad odesílání a přijímání telegramů při nastavování skeneru je znázorněn po krocích v tabulce níže.[13]

Krok	Akce	Pozn.
1	Spuštění LMS (power-on).	
2	LMS291 posílá „power on“ řetězec.	Pouze během spouštění
3	LMS je odeslán příkaz na přepnutí do instalačního módu.	Příkaz 20h
4	LMS odpovídá potvrzením (Acknowledge).	06h
5	LMS odpovídá na příkaz přepnutí.	Odpověď A0h
6	Příkaz pro nastavení parametrů je odeslán do LMS	Obvykle příkaz 77h
7	LMS odpovídá potvrzením (Acknowledge).	06h
8	LMS odpovídá potvrzením „parametr úspěšně změněn“.	Odpověď F7h
9	Příkaz pro přepnutí do měřícího módu je odeslán do LMS	Příkaz 20h
10	LMS odpovídá potvrzením (Acknowledge).	06h
11	LMS odpovídá potvrzením „mód úspěšně změněn“.	Odpověď A0h
12	Čekání na další příkaz, např. start přenosu dat.	Začátek další akce

Tab. 5.: Příklad komunikace mezi LMS a PC

4.2 Navázaní komunikace s LMS

Jakmile je otevřeno spojení pomocí FTDI USB převodníku lze začít komunikaci se skenerem. LMS 291 po spuštění automaticky odpovídá *spouštěcím* řetězcem (90h). Řetězec obsahuje typ jednotky a verzi systémového softwaru (např. LMS291; 301063; V02.10). Po spuštění je LMS připraveno k další činnosti přibližně po 60 sekundách. Příkazy přijaté během této rozběhové doby jsou ignorovány.

Popis	STX	Adresa	Délka		Odpověď	Data		CRC	
						Data	Status		
Pozice bytu	1	2	3	4	5	6 až 26	27	28	29
Hex. hodnota	02	80	17	00	90	02 80 17 00 90 4C 4D 53 32 30 30 3B 33 30 31 30 36 33 3B 56 30 32 2E 31 30 20 10 72 D0	10	72	D0

Tab. 6.: Telegramu spouštěcí zprávy LMS

4.3 Základní nastavení LMS

Po spuštění a navázání komunikace je žádoucí zjistit aktuální nastavené hodnoty LMS. K tomu slouží tzv. *status* příkaz. Odpovědí na tento příkaz je 162 bytů dlouhý telegram obsahující ve své datové části aktuální nastavení všech parametrů. Standardní nastavení je přenosová rychlost 9 600 Bd, úhlový rozsah 180° po 1° a měření v cm.

Všechny tyto vlastnosti lze následně změnit a každý z těchto procesů parametrizace popsaných níže má za následek zapisovací cyklus do EPROM LMS. Jakmile jsou parametry uloženy v EPROM, nemusí být konfigurace prováděna po každém spuštění. Nastavit se musí pouze komunikační parametry jako např. přenosová rychlost.[13]

4.4 Změna přenosové rychlosti

Po každém spuštění LMS je přenosová rychlost resetována na 9600 baudů. Pro přístup k měřeným datům při vyšších rychlostech je nutné rychlost přednastavit po každém spuštění LMS jedním z následujících příkazů.

Přenosová rychlost	Telegram nastavení PC -> LMS	Telegram odpovědi LMS -> PC
9 600 Baud	02 00 02 00 20 42 52 08	06 02 81 03 00 A0 00 10 36 1A
19 200 Baud	02 00 02 00 20 41 51 08	06 02 81 03 00 A0 00 10 36 1A
38 400 Baud	02 00 02 00 20 40 50 08	06 02 81 03 00 A0 00 10 36 1A
500 kBaud	02 00 02 00 20 48 58 08	06 02 81 03 00 A0 00 10 36 1A

Při přenastavování přenosové rychlosti jedním z příkazů uvedených v tabulce dojde k přerušení spojení mezi LMS a PC. Pro obnovení spojení je nutné přednastavit přenosovou rychlost také na USB převodníku.

Jak již bylo zmíněno v předcházejících kapitolách, pouze s rychlostí nastavenou na 500 kBaudu nedochází k žádným ztrátám dat. Při nižších rychlostech může docházet ke ztrátám naměřených hodnot podle následující tabulky.

Přenosová rychlost	Rozlišení		
	0.25°	0.5°	1°
9 600 Bd	16	29	30
19 200 Bd	8	15	15
38 400 Bd	4	8	8
500 kBd	0	0	0

Tab. 7.: Vztah přenosových rychlostí a ztrát dat.[13]

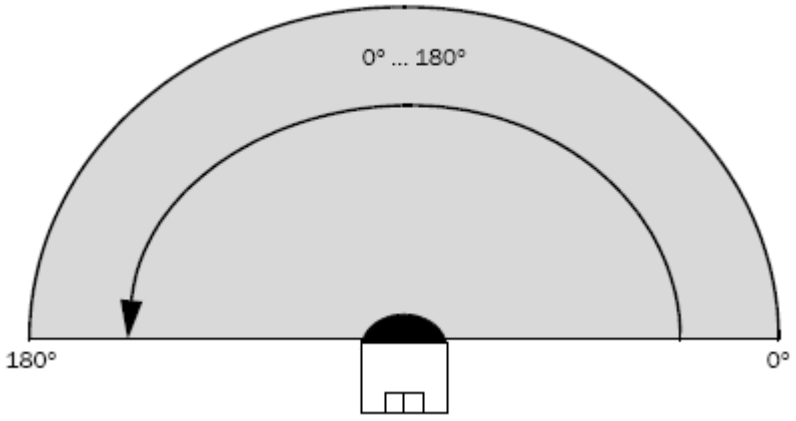
4.5 Volba měřícího módu

LMS může poskytovat měřená data při úhlovém rozsahu 0 až 100° nebo 0 až 180° a při úhlovém rozlišení 1°, 0.5° nebo 0.25° (pouze pro 100°). Tyto hodnoty lze opět nastavit posláním příslušného příkazu z následující tabulky.

Měřící mód			Telegram nastavení PC -> LMS	Telegram odpovědi LMS -> PC
Rozsah	Rozlišení	Počet hodnot		
0°-100°	1°	101	020005003B640064001D0F	0602800700BB0164006400104FBD
0°-100°	0.5°	201	020005003B64003200B159	0602800700BB0164003200101710
0°-100°	0.25°	401	020005003B64001900E772	0602800700BB016400190010BB46
0°-180°	1°	181	020005003BB40064009749	0602800700BB01B4006400105B30
0°-180°	0.5°	361	020005003BB40032003B1F	0602800700BB01B400320010039D

Tab. 8.: Měřící módy

Při nastaveném rozsahu 0° až 100° je nutno v dalším zpracování dat brát v potaz, že první naměřená hodnota odpovídá 40° v příslušném zorném poli skeneru a poslední hodnota odpovídá 140°. Měření probíhá proti směru hodinových ručiček.



Obr. 11.: Rozsah měření LMS

V závislosti na prostředí lze také nastavit jednotky měřených hodnot na *cm* nebo *mm*. Měření v milimetrovém módu znamená větší přesnost, ale menší dosah. Ten by však

měl být ve většině měření uvnitř budov dostatečný. Přenastavení jednotek je patrně časově nejnáročnější změnou parametru LMS a může zabrat až 7 sekund. Příkazy přijaté během této doby jsou ignorovány.

4.6 Měření dat

Jakmile jsou nastaveny všechny potřebné parametry LMS, zbývá nastavit vertikální rozsah skenování ovládaný servomotorem a poté spustit měření. Na rozdíl od klasických aplikací získávání dat, je výstupem LMS kontinuální proud dat, které je nutné průběžně zpracovávat. LMS provádí měření a posílá data, dokud mu není poslán příkaz zastavení.

Při měření je kontinuální proud dat v pravidelných intervalech načítán aplikací. Jakmile dojde k detekci hlavičky telegramu a v zásobníku je uložen dostatečný počet znaků pro celý telegram tak je tento zpracován. Telegram se rozdělí na jednotlivé části podle tabulky v odstavci 4.1, kde nejdůležitější části jsou samozřejmě naměřená data vzdáleností. Průběžně jsou také získávány hodnoty vertikálního natočení skeneru, které jsou následně přiřazeny jednotlivým datům. Jakmile je hodnota natočení skeneru rovna koncové hodnotě nastaveného vertikálního rozsahu je do LMS poslán příkaz pro zastavení měření.

Každý zpracovaný telegram ve své datové části obsahuje daný počet naměřených hodnot podle nastaveného rozlišení. Pro každou naměřenou hodnotu jsou výstupem dva data byty popsané v tabulce níže.

	Více důležitý data byte								Méně důležitý data byte							
Číslo Bitu	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Binární hodnota	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
Hex. hodnota	00 až FF								00 až FF							
Dec. hodnota	0 až 65 535															

Tab. 9.: Formát kódování hodnot vzdáleností

Při standardní konfiguraci jsou pro reprezentaci vzdálenosti používány data bity 0 až 12. Těchto 13 bitů umožňuje vyjádřit $2^{13} - 1 = 8191$ hodnot. V závislosti na zvoleném rozlišení mohou být reprezentovány různé měřící rozsahy. Pokud je zvoleno rozlišení 1 mm tak lze teoreticky naměřit maximální vzdálenost 8,191 m a při rozlišení 1 cm to může být až 81,91 m. Hodnoty vzdáleností jsou zaznamenávány stejně jako celý telegram v hexadecimálním formátu. [3]

4.7 Výstup dat

Naměřená data se ukládají do textového souboru. Textový formát má sice obvykle větší velikost než by měla jeho binární obdoba, ale zato umožňuje snadné čtení a kontrolu dat uživatelem. V tomto souboru je uložen celkový počet provedených měření a pro každé jednotlivé měření jsou pak uloženy celé přijaté telegramy jednotlivých skenů rovin. Pro každou rovinu je uložen vertikální úhel, při kterém byla naměřena, díky němuž lze následně vypočítat souřadnice jednotlivých bodů v prostoru.

5 NAVRŽENÉ ŘEŠENÍ PRO ZPRACOVÁNÍ DAT

Druhou částí je aplikace implementující několik základních metod vytváření 3D mapy okolí z naskenovaných dat. Použití 3D map v oblasti mobilních robotů může mít dvojí účel. Prvním z nich je reprezentace naskenovaného prostředí pro potřeby člověka. Obecně se může jednat např. o úkoly mapování člověku nepřístupných míst, jako jsou opuštěné doly, požkozené budovy či nebezpečná prostředí. V těchto algoritmech jde především o co nejdokonalejší rekonstrukci prostředí. Opakem jsou metody vytváření map pro potřeby mobilních robotů, kterých je mnoho druhů. U většiny z nich však jde především o co nejjednoduší reprezentaci okolí, která umožňuje rychlou a snadnou navigaci a lokalizaci.

Ať už z jedné nebo druhé kategorie je v současné literatuře popsáno velké množství algoritmů používajících různé přístupy vedoucí k různým výsledkům. Mnoho z nich je značně komplikovaných a postaveno na složitých matematických metodách. Algoritmy použité v této práci patří k základním představitelům různých typů přístupů a mohou tak sloužit k jejich porovnání.

5.1 Vstup a reprezentace dat

Vstupem aplikace jsou data uložená v podobě telegramů spolu s jejich vertikálním úhlem β, při kterém byla naměřena. Pro další vizualizaci je potřeba každou naměřenou hodnotu převést do pravotočivého kartézského souřadného systému. Z telegramu lze vyčíst počet jednotlivých naměřených hodnot a díky tomu určit v jakém úhlovém rozsahu bylo provedeno měření. Znalost úhlového rozsahu umožňuje každé naměřené hodnotě vzdálenosti přiřadit její horizontální úhel α.

Obr. 12.: Úhly, při kterých byla naměřena hodnota vzdálenosti

Z načteného vertikálního úhlu a přiřazeného horizontálního úhlu spolu se vzdáleností lze každou naměřenou hodnotu reprezentovat jako bod v prostoru o souřadnicích *x, y, z*, které se vypočítají z následujících vztahů:

$$x = \cos\alpha * vzdálenost \tag{5.1}$$

$$y = \sin\alpha * \sin\beta * vzdálenost \tag{5.2}$$

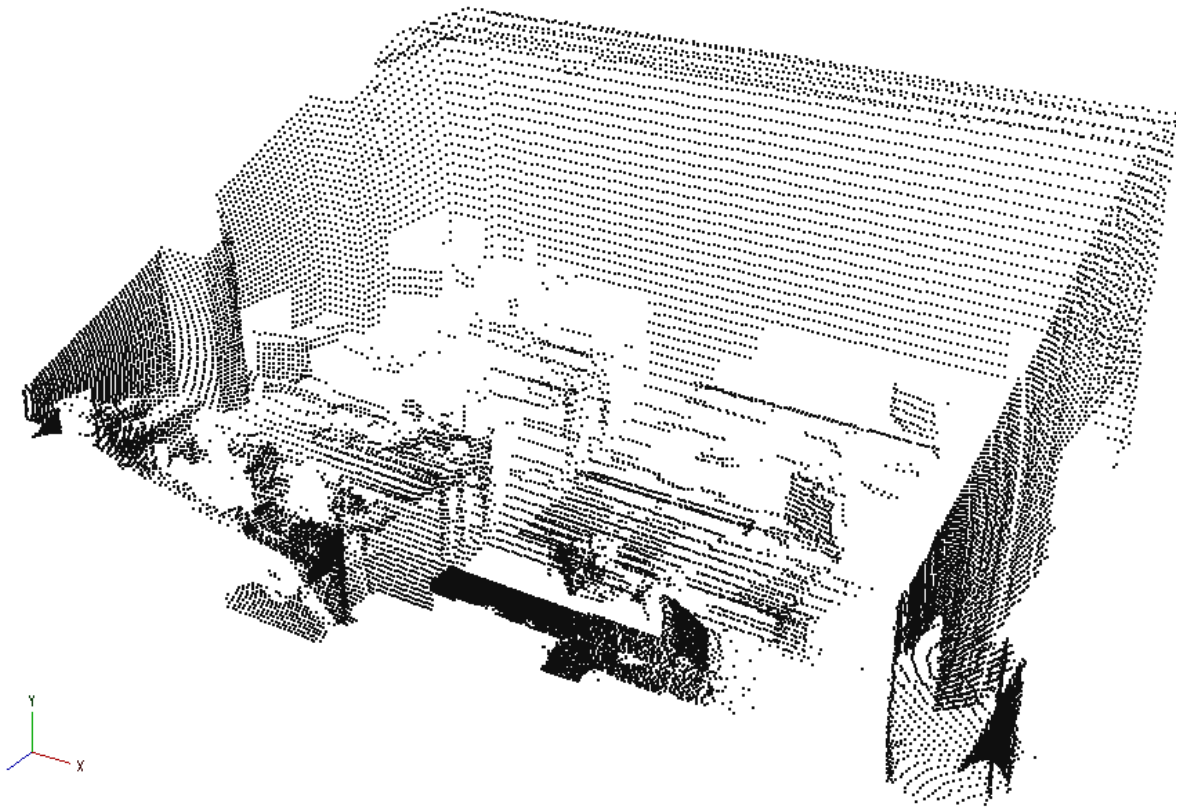
$$z = -\sin\alpha * \cos\beta * vzdálenost \tag{5.3}$$

Po zpracování všech telegramů dojde k vytvoření tzv. *mračna bodů*. Naskenované mračno bodů může podle nastavených parametrů obsahovat desítky až stovky tisíc bodů.

Jako experimentální měření bylo provedeno skenování místnosti laboratoře. Skenování proběhlo v režimu horizontálního pásma 180° - 0.5° a ve vertikálním pásmu v rozsahu 80° po 1° kroku. Výsledkem bylo mračno 29 765 bodů.

Obr. 13.: Laboratoř z pohledu skeneru [21]

Mračno bodů je díky svoji struktůře a velkému množství bodů samo o sobě vhodné pouze pro základní reprezentaci naskenovaných dat. Z tohoto důvodu je nutné jej převést do jednodušších struktur. Metod zpracování mračna bodů je velké množství lišících se svým účelem. V zeměměřičství a strojírenství se obvykle používají metody pro převod do vektorových kreseb a CAD formátů. V robotice se používají různé experimentální metody rekonstrukce ploch a detekce objektů, které se odvíjejí od konkrétních problémů. V tomto případě jsou popsány algoritmy vhodné pro účely mapování okolí pro navigaci a lokalizaci mobilních robotů.



Obr. 14.: Naskenované mračno bodů (přibližně 30 000 bodů)

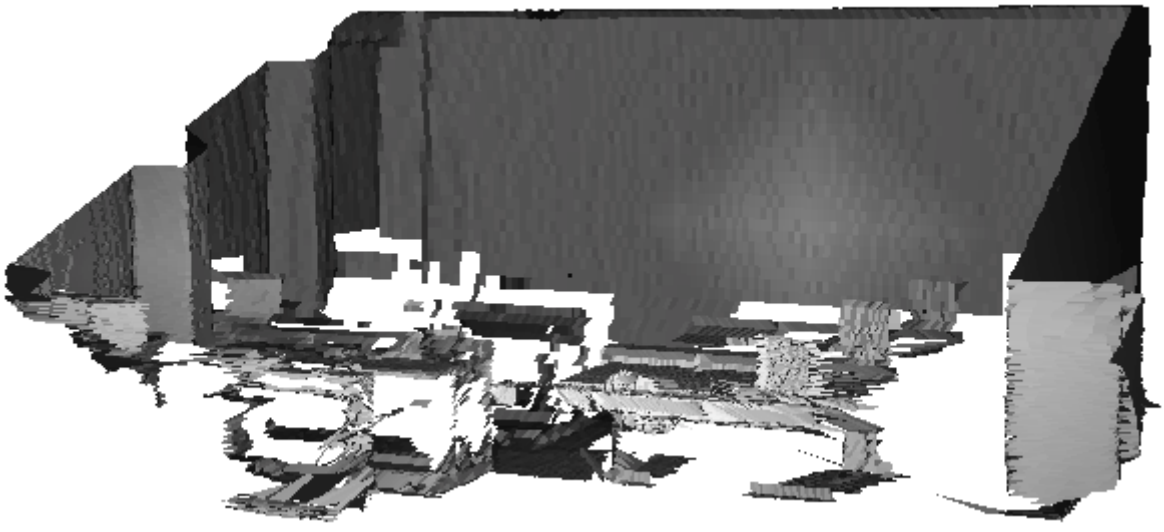
5.2 Rekonstrukce povrchů

První z implementovaných metod 3D mapování spadá do oblasti úloh pro potřeby člověka. Jejím cílem je tedy co nejdokonalejší rekonstrukce skutečného prostředí. Metod rekonstrukce povrchů (též strukturální mapování) z mračna bodů je celá řada, a netýkají se pouze oblasti robotiky. Ze známých matematických přístupů lze jmenovat např. Delaunyho sítě, či Voroniyovi diagramy.

Zde použitý algoritmus vychází z prací [5] [8] a poskytuje srovnatelné výsledky s výše zmíněnými metodami triangulce. Je však značně méně komplikovaný, protože chytře využívá charakteru vstupních dat, kde jsou jednotlivé body řazeny za sebou stejně jako jednotlivé roviny skenování.

Mapa je vytvořena hledáním sousedních bodů dvou sousedních rovin měření. Algoritmus postupně v každém kroku vybere čtyři body $z_{\tau}^{[i]}, z_{\tau}^{[i+1]}, z_{\tau+1}^{[i]}, z_{\tau+1}^{[i+1]}$, kde τ označuje rovinu měření a i index bodu ležícího v této rovině. Pokud jsou tyto body v prostoru od sebe vzdáleny méně než je stanovená hodnota δ, tak lze tvrdiť, že tvoří malou plochu. Pro tuto plochu je posléze vytvořen obdélníkový polygon s vrcholy odpovídají těmto čtyřem hodnotám a je přidán do mapy. Test na vzdálenost mezi body v prostoru je nezbytný, jelikož velké nespojitosti korespondují s volným prostorem (např. otevřené dveře, okno).

Algoritmus aplikovaný na testovací mračno bodů je znázorněn na obrázku. Je dobře patrná dobrá rekonstrukce povrchu stěn i menších objektů jako jsou monitory. Tento přístup je vhodnou reprezentací naskenovaného mračna bodů pro člověka. Pro potřeby mobilních robotů však už moc vhodnou reprezentací není, jelikož se skládá z velkého množství malých plošek, které sice poskytují dobrý detail, ale ten je pro navigaci a lokalizaci zbytečný a příliš by tyto procesy zatěžoval.



Obr. 15.: Aplikace strukturálního mapování na mračno bodů

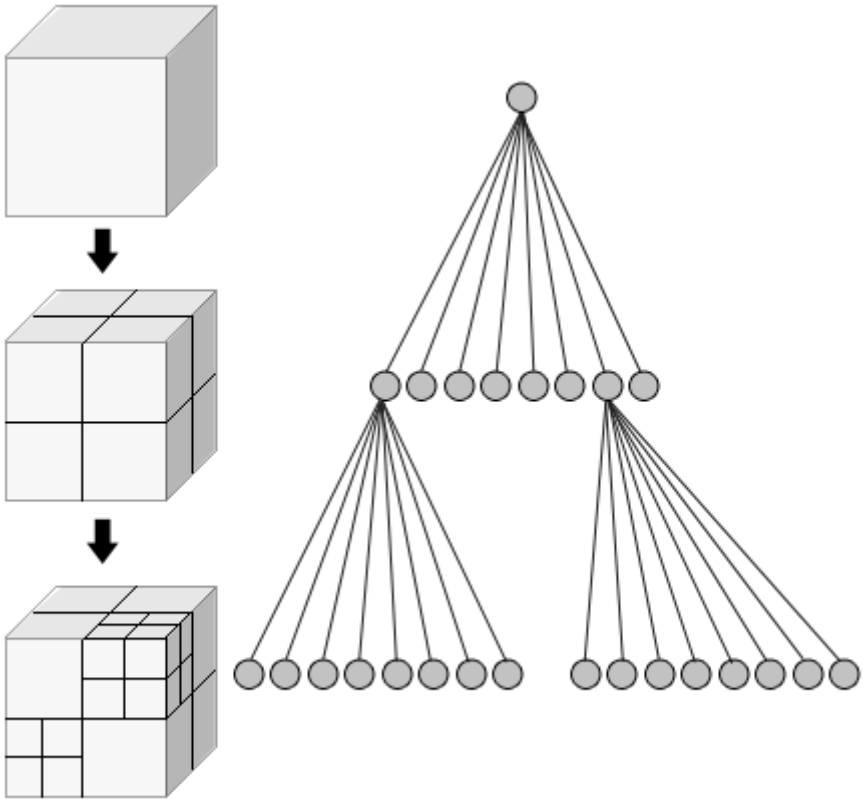
Jako velmi dobrou aplikaci této metody lze úvést také několik prací S. Thruna např. [6] [7] popsanou v odstavci 5.7.

5.3 Hierarchické dělení prostoru pomocí oktalového stromu

Hierarchické metody dělení prostoru představují způsob, jak systematicky rozdělit prostor na disjunktní oblasti (tzv. *buňky*). Dělení se provádí pomocí dělících (řezných) rovin, které tvoří hranice mezi buňkami. Každý bod prostoru pak jednoznačně do nějaké buňky patří. Jedná se tak v podstatě o 3D obdobu velmi oblíbeného 2D mapovacího algoritmu mřížky obsazenosti.

Hierarchie dělení má strukturu stromu budovaného shora dolů, jehož každý uzel reprezentuje jednu buňku resp. ty objekty, které se v ní nachází. Maximální počet dceřiných uzlů je dán vztahem 2^n, kde n je počet řezných rovin. Podmínkou pro rozdělení buňky nebo naopak zastavení dělení může být např. dosažená hloubka stromu. Informace o objektech jsou uloženy v listech stromu v podobě odkazů.

Oktalový strom (také *oktanový strom, octree*) je hierarchická stromová datová struktura založená na postupném dělení prostoru třemi rovinami kolmými na souřadnicové osy. Stavba stromu začíná vytvořením kořenového uzlu – krychle, která obsahuje všechny body objektu. Poté dojde k rozdělení prostoru pomocí dělících rovin, které jsou umístěny ve středu rozdělované buňky. Dělením vzniká vždy osm buněk (tzv. *oktanů*). Pokud buňka obsahuje některé body svého předchůdce je přidána do grafu a od předchůdce převezme patřičné body náležící do jejího objemu. V opačném případě je z dalšího zpracování vypuštěna, dojde k oříznutí větve. Každý uzel tak může mít nula až osm potomků. Uzel bez potomků se nazývá list a jako jediný typ uzlů z celého stromu obsahuje odkazy na objekty, které se nacházejí uvnitř prostoru ohraničeného tímto listem.



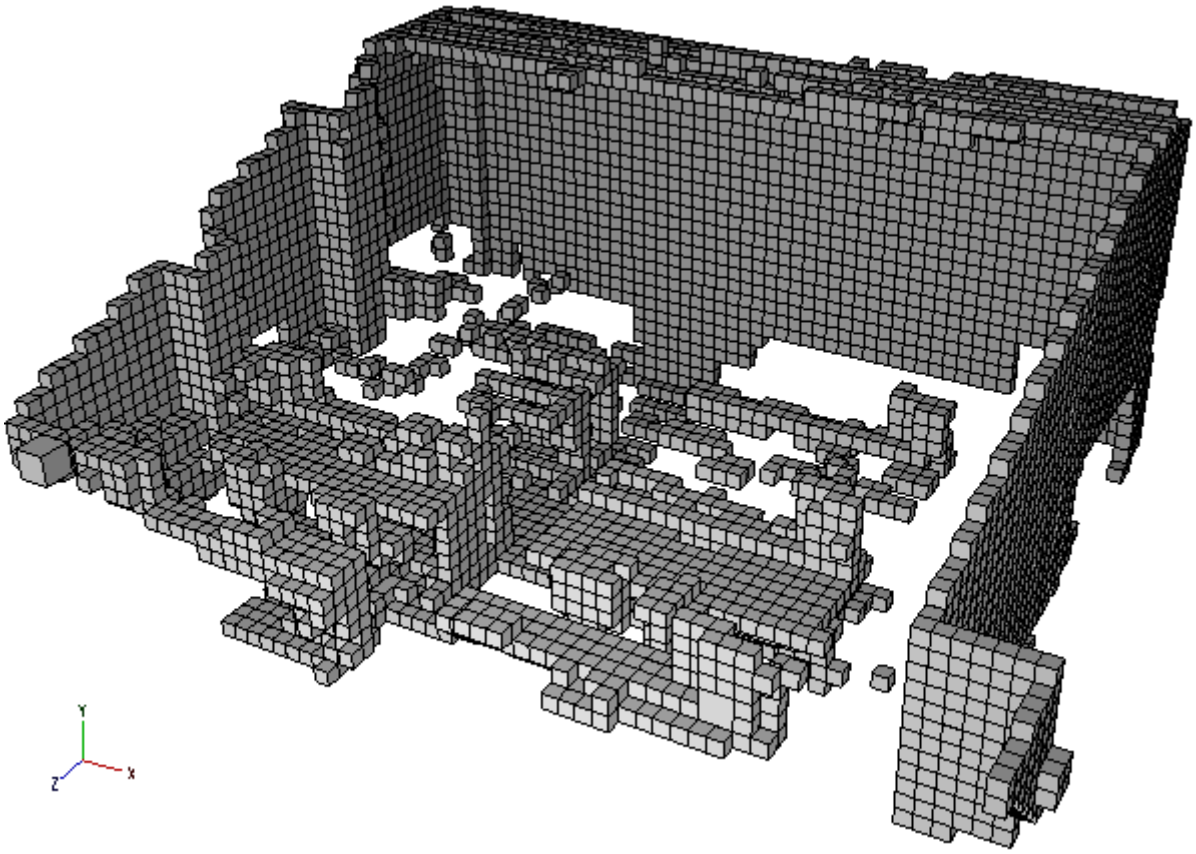
Obr. 16.: Princip oktalového stromu

Každý uzel je definován svým středem (bod v prostoru, kterým vedou dělící roviny) a svou dimenzí (velikost hrany krychle ohraničené daným uzlem), přičemž všechny buňky na stejné úrovni mají shodnou dimenzi. Platí vztah, že dimenze dceřiného uzlu je vždy rovna polovině dimenze rodičovského uzlu.

Pro zastavení dělení buněk lze v případě oktalového stromu použít několika kriterií, jejichž vhodnost se odvíjí od řešeného problému. V tomto případě je jediným vhodným způsobem ukončení dělení dosažená maximální hloubka. Tím je zajištěno, že všechny listy mají stejnou velikost, aniž by záleželo na tom kolik obsahují bodů. Implementace umožňuje také zadat místo maximální hloubky, maximální velikost hrany listů (jejich dimenzi), ze které je následně vypočtena potřebná hloubka stromu.

Na Obr.17. je znázorněn příklad použití dělení pomocí oktalového stromu na testované mračno bodů. Oktalový strom hloubky šest pro dostatečně jemné rozdělení vyprodukoval přibližně 5 000 segmentů.

Hlavní přednostní oktalového stromu je nevelká výpočetní náročnost a jeho snadné využití pro navigaci robotu jakožto mřížky obsazenosti.



Obr. 17.: Příklad aplikace oktalového stromu na mračno bodů

5.4 Segmentace pomocí Houghovy transformace

Jiným druhem algoritmu je tvorba objektových (též geometrických) map skládajících se ze základních geometrických tvarů, jako úsečky, plochy, tělesa atd. Objektové mapy mají vůči mřížkovým mapám čtyři jednoduché výhody. Objektové mapy mohou být více kompaktní, zvláště v dobře strukturovaných prostředích. Zadruhé, mohou být více přesné, pokud uvažujeme, že použitá primitiva mají adekvátní geometrii popisující

skutečné objekty v prostředí. Zatřetí, objektová reprezentace se zdá nezbytnou pro popis dynamických prostředí, kde se v průběhu času mohou pozice objektů měnit. A začtvrté, objektové mapy jsou obvykle blíže k lidskému vnímání okolí než mřížkové mapy obsazenosti. Avšak objektové mapy trpí také velkou nevýhodou a to, že jejich kvalita je obvykle omezena na prostředí, které lze vyjádřit jednoduchými geometrickými reprezentacemi.

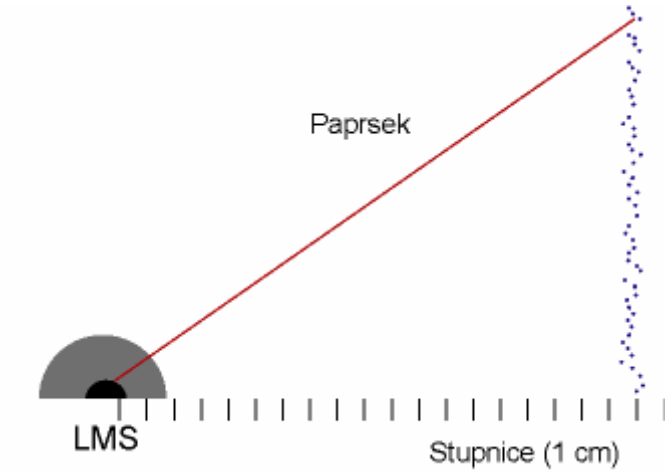
Mnou zvolená konkrétní implementace objektové mapy vychází z prací [1][2]. Celý algoritmus se skládá z několika dílčích podúloh. Prvním krokem je předzpracování dat, následuje detekce úseček pomocí Houghovy transformace. Ty jsou následně využity pro extrahování ploch a jejich segmentaci v objekty.

5.4.1 Předzpracování dat

Kvalita detekovaných úseček pomocí metody popsané níže se odvíjí od kvality vstupních dat. Hrubá naskenovaná data v některých místech obvykle trpí značnými nepřesnostmi. Měření je zatíženo systematickou chybou, jejíž velikost závisí na zvolených jednotkách měření. Pro meření v milimetrech, kde je systematická chyba menší udává výrobce hodnotu max. ± 15 mm. Měření rovinné plochy zatížené touto chybou je znázorněno na obrázku níže. Další vlastností měření je, že skener emituje laserový paprsek ve sférickém smyslu, takže hustota bodů blíže ke zdroji je mnohem větší apod. Tudíž je žádoucí data nejprve předzpracovat. Pro tyto účely jsou implementovány dvě metody.

První z nich je modifikace oktalového stromu popsaného výše. Nejprve dojde ke konstrukci celého stromu, podle zadané maximální hloubky. Každý list stromu je poté nahrazen jedním bodem. Tím dojde nejen k redukci, ale také k zarovnání bodů do přímek. Nevýhodou této metody je nepříliš vhodné nahrazování diagonálních ploch skokovými přechody.

Druhou metodou je jednoduché porovnávání vzdáleností mezi sousedními body. Pokud je vzdálenost mezi dvěma sousedními menší než zadaná hodnota, tak jsou tyto dva body spojeny v jeden. Takto lze dosáhnout konstantní hustoty bodů ve všech vzdálenost od skeneru. Kvalita zarovnání není tak vysoká jako u předchozí metody, ale odpadá zde problém s diagonálními plochami.



Obr. 18.: Systematická chyba LMS

5.4.2 Detekce úseček pomocí Houghovy transformace

Houghova transformace je technika používaná především v analýze a digitálním zpracování obrazu. Cílem této metody je najít nedokonalé instance objektů z definované třídy tvarů. Proces hledání se uskutečňuje v parametrickém prostoru, ze kterého jsou kandidáti na objekty získáváni jako lokální maxima v takzvaném Houghovu prostoru, který je jednoznačně definován pomocí výpočtu Houghovy transformace.

Klasická Houghova transformace je používána především k identifikaci přímek v obrazu. Byla objevena Richardem Dudou a Peterem Hartem v roce 1972, kteří ji nazvali "generalizovaná Houghova transformace" vycházející z patentu Paula Hougha.

V automatizované analýze digitálního obrazu obvykle vystává problém detekce jednoduchých tvarů, jako jsou přímky, kruhy nebo elipsy. V tomto konkrétním případě se bude jednat pouze o lineární Houghovu transformaci pro detekci přímek, proto se zde nebudu zabývat problémy s detekcí složitějších tvarů.

Díky nepřesnostem at už obecně obrazových dat, nebo v tomto konkrétním případě v procesu skenování dat, vznikají chybějící body, prostorové odchylky od ideálních přímek nebo šum v získaných datech. Vzhledem k těmto problémům nastává netriviální úloha přiřazování konkrétních bodů k jednotlivým přímkám. Houghova transformace řeší tento problém pomocí explicitního procesu volby nad sadou parametrizovaných objektů.

Jak již bylo řečeno nejjednodušším případem lineární transformace je detekce přímek. Přímka může být definována rovnicí jako $y = kx + q$ a může být graficky znázorněna pro každý bod definovaný souřadnicemi *(x, y)*. Hlavní myšlenkou je neuvažovat přímku jako spojnici bodů definovaných souřadnicemi *(x, y)* ale pomocí její směrnice k a úseku q. Pokud vyjdeme z tohoto faktu tak můžeme přímku definovat také jako bod *(k, q)* v parametrickém prostoru. Avšak zde také vystává problém s vertikálními přímkami, kde mohou neomezeně narůstat hodnoty k a q. Proto se v Houghově transformaci používají dva jiné parametry obvykle označované jako r a θ (theta). [18]

Parametr r reprezentuje délku kolmice k přímce vedené z počátku souřadného systému a θ (theta) reprezentuje úhel mezi touto kolmicí a vodorovnou osou souřadného systému. Pomocí těchto veličin můžeme rovnici přímky zapsat jako:

$$y = \left(-\frac{\cos\theta}{\sin\theta} \right) x + \left(\frac{r}{\sin\theta} \right) \tag{5.4}$$

Nebo zjednodušeně jako

$$r = x\cos\theta + y\sin\theta \tag{5.5}$$

Tudíž je možné přiřadit každé přímce dvojici *(r, θ)*, která je jedinečná pokud jsou splněny podmínky $\theta \in [0, \pi],\ r \in R$ nebo $\theta \in [0, 2\pi],\ r \geq 0$.

Rovina *(r, θ)* pro sadu přímek je obvykle označována jako Houghuv prostor. Každým bodem v rovině může procházet nekonečný počet přímek. Pokud má bod souřadnice *(x, y)*, tak všechny přímky procházející tímto bodem splňují rovnici 2.2. Ta odpovídá sinusové křivce v rovině *(r,θ)*, která je jedinečná pro tento bod. Pokud se křivky odpovídající dvěma bodům v Houghově prostoru protnou, tak pozice tohoto překrytí udává parametry *(r,θ)* přímky procházející oběma těmito body. Obecněji množina bodů tvořící přímku vytvoří sinusoidy, které se protnou v souřadnicích daných parametry této přímky. Tedy problém hledání bodů ležících na jedné přímce je zde převeden na problém hledání lokálních maxim Houghova prostoru překrývajících se křivek. [18]

Algoritmus Houghovy transformace používá pole (takzvaný *akumulátor*), pro ukládání nalezených přímek. Rozměry tohoto pole jsou dány počty neznámých parametrů

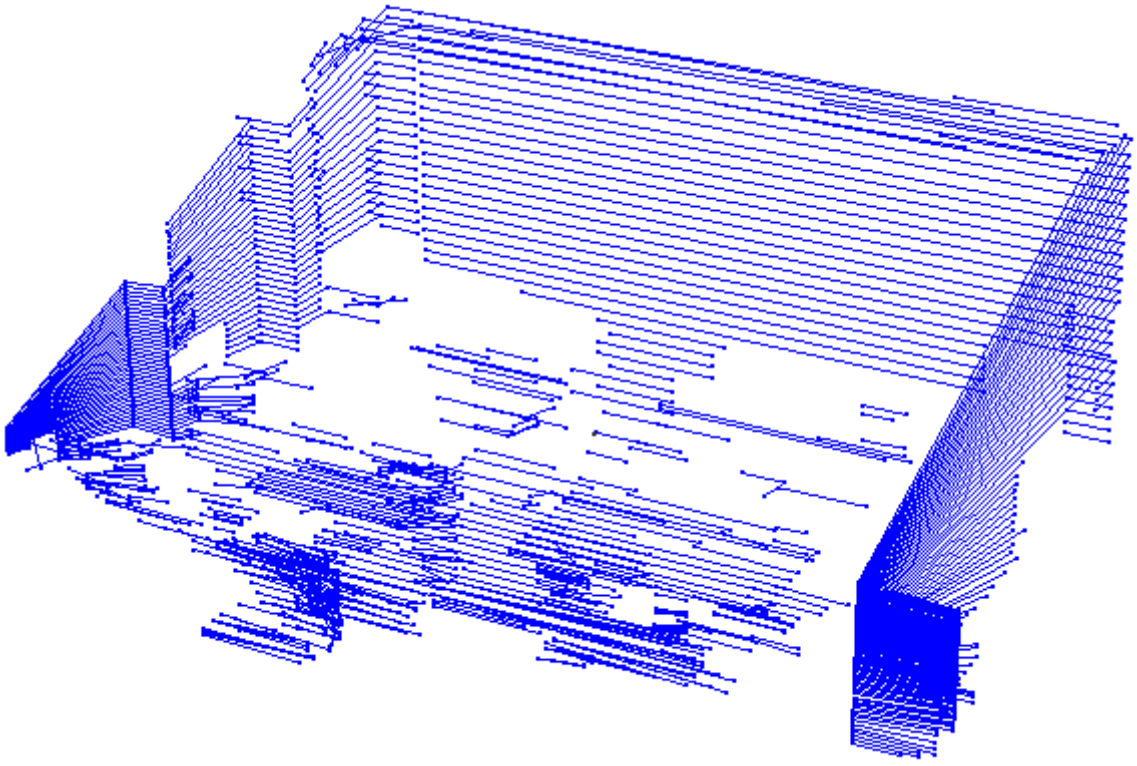
Houghova problému. V tomto lineárním případě tedy dvou parametrů r a θ. Algoritmus pro každý bod (z množiny dat) vypočítá parametry všech přímek v zadané množině úhlů procházejících tímto bodem.

Při obvyklém řešení se v pozici *akumulátoru* odpovídající těmto parametrům pouze inkrementuje počet bodů na přímce. Následuje procházení akumulátoru a hledání lokálních maxim podle zadané prahové hodnoty. U takto získaných přímek známe pouze jejich geometrické definice, ale neznáme informace o jejich délce, a tudíž vzniká další problém, jehož podstatou je zjišťování jaký jejich úsek naleží datům.

Tento problém je vyřešen díky charakteru skenovaných dat. Do pozic v akumulátoru nejsou vkládány pouze počty bodů na přímce, ale přímo ukazatele na tyto body. Při skenování jsou jednotlivé body získávány postupně proti směru hodinových ručiček. V tomto pořadí jsou také vkládány do akumulátoru a tak lze snadno zjistit počátek úsečky jako první bod a její konec jako poslední bod obsažený v pozici akumulátoru dané parametry přímky.

Pro větší flexibilitu detekce může uživatel ovlivnit většinu parametrů vyskytujících se v této metodě. Lze zadat např. minimální počet bodů, které tvoří úsečku, maximální vzdálenost mezi dvěma sousedními body, minimální délku úsečky, nebo úhlový krok, ve kterém se mají hledat úsečky.

Obrázek níže ukazuje aplikování detekce úseček na testovací mračno bodů, které bylo předzpracováno oktalovým stromem. Počet detekovaných úseček v tomto případě činí 876.

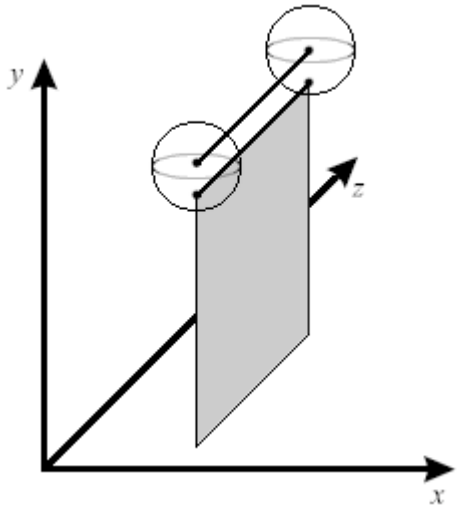


Obr. 19.: Příklad detekce úseček aplikované na předzpracované mračno bodů pomocí oktalového stromu

5.4.3 Detekce ploch

Předcházející metoda aproximuje naskenovanou rovinnou plochu sadou úseček. Úkolem je rozeznat takovéto struktury ve vstupních datech a spojit tyto jednotlivé úsečky v jeden celek tvořící plochu. Pro dosažení tohoto cíle algoritmus detekce ploch provádí následující kroky:

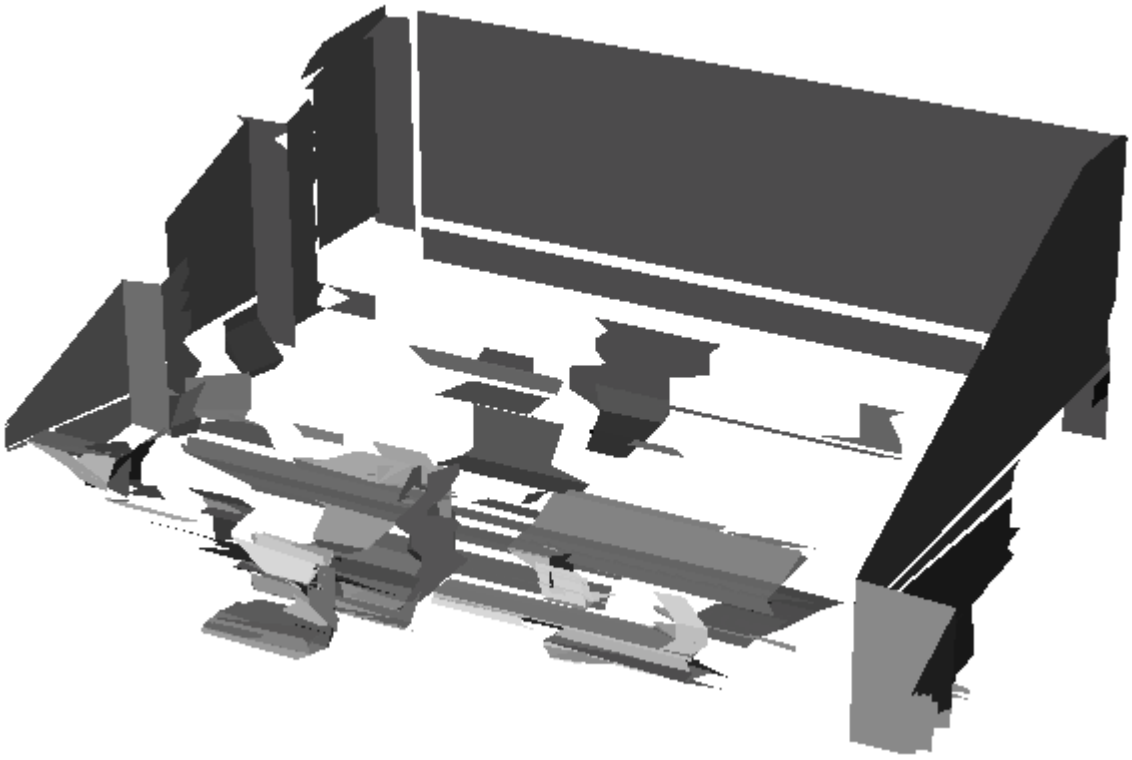
1. Nejprve je uložena sada úseček z první naskenované roviny.
2. Každá následující úsečka je porovnávána se sadou uložených úseček. Pokud je nalezena vhodná dvojice úseček pak jsou tyto dvě transformovány v plochu.
3. Pokud není nalezena takováto vhodná dvojice, může být úsečka rozšířením již nalezené plochy. V tomto případě se úsečka porovnává s horní hranou (neboli poslední přidanou úsečkou) plochy, a pokud jsou splněny podmínky je horní hrana plochy nahrazena testovanou úsečkou čímž dojde k rozšíření plochy.
4. Pokud není splněn ani jeden ze dvou předcházejících kroků je úsečka vložena do sady zmíněné v kroku 1.



Obr. 20.: Princip rozšíření plochy o novou úsečku

Pro spojení dvojice úseček musí být splněna tři kritéria. První kritérium vyžaduje, aby koncové body úseček od sebe neležely ve větší vzdálenosti, než je předem zadaná hodnota. Druhé kritérium vyžaduje, aby úhel θ (viz. Houghova transformace) mezi dvojicí úseček byl menší než zadaná hodnota δ. Druhý předpoklad je nezbytný např. pro správné vyhodnocování množiny krátkých úseček, kde může být první předpoklad splněn velmi snadno. Třetím kritériem je shodnost normály nově rozšířené a původní plochy.

Aplikováním detekce ploch na množinu detekovaných úseček z předchazejícího kroku vzniklo 238 samostatných ploch. Výsledek je znazorněň na obrázku níže.



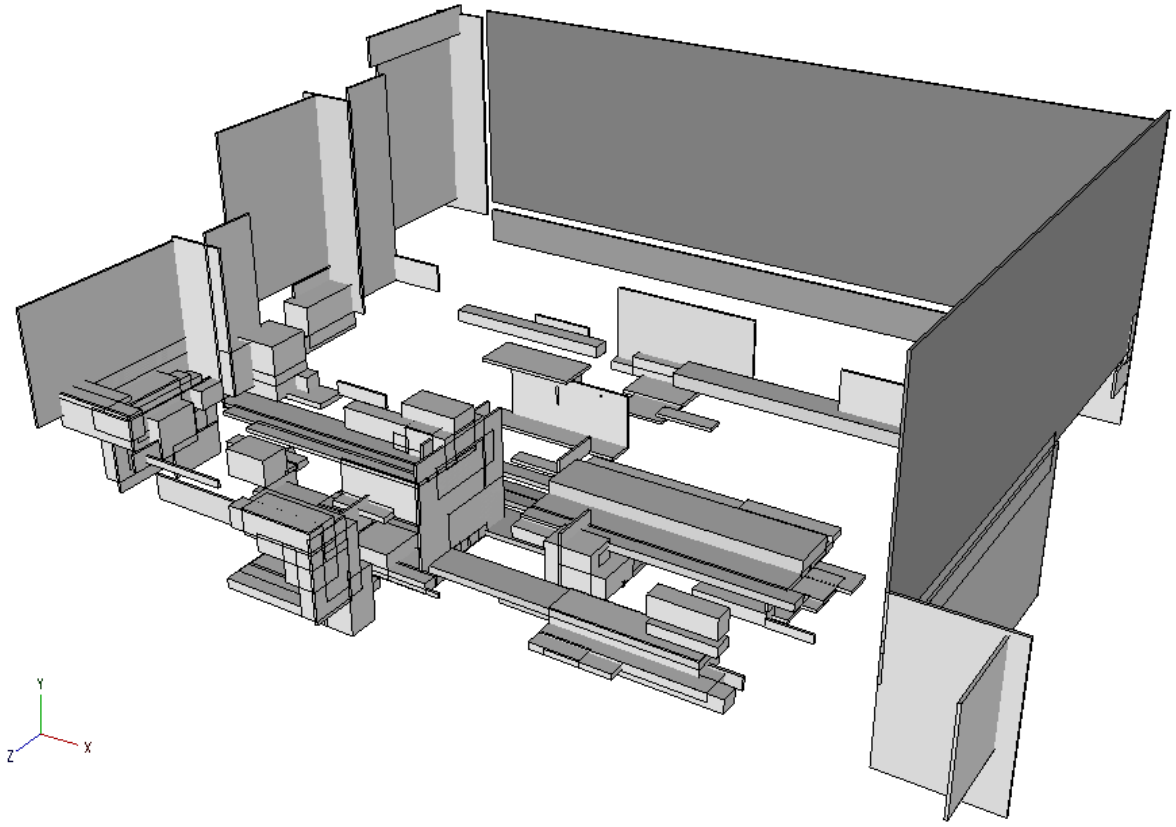
Obr. 21.: Plochy vytvořené z detekovaných úseček

5.4.4 Segmentace objektů

Pro získání konečných objektů zbývá aplikovat vhodnou metodu segmentace. V [1] je popsána metoda, která v prvním kroku algoritmu vybere plochy s největším obsahem, kolem kterých je následně opsán kvádr a kolem kvádru pomyslná koule. Opsaný kvádr poskytuje objemovou reprezentaci plochy - segment. V dalších krocích algoritmu se pro každý detekovaný segment testují všechny zbývající plochy, zda jsou dostatečně blízko segmentu, neboli zda jsou pojmuty v opsané kouli segmentu. Pokud tuto podmínku splňují, jsou segmentem pohlceny, čímž dojde k jeho zvětšení. V prezentovaných výsledcích práce [1] je však na první pohled patrná nedokonalost tohoto algoritmu především v tom, že ne všechny detekované plochy musí být ve výsledku součástí některého ze segmentů. To by mohlo vést k hrubým chybám při navigaci v takto nedokonale zmapovaném 3D prostředí.

V této práci zvolený přístup segmentace se odvíjí od výše zmíněného algoritmu, ale je přímočařejší a odstraňuje původní nedokonalost za cenu většího počtu konečných segmentů. V prvním kroku je opsán kvádr kolem každé z detekovaných ploch. Jelikož je při tomto přístupu často opsaný kvádr menší plochy zcela pohlcen v objemu opsaného kvádru větší plochy jsou tyto menší kvádry odstraněny v dalším kroku. Redukce probíhá v cyklu tak, že segmenty jsou seřazeny od největšího po nejmenší. Nejprve se zvolí první segment a testuje všechny ostatní, zda-li jsou pohlceny v jeho objemu s nastavitelnou tolerancí δ. Pokud ano, tak je pohlcený segment zcela vypuštěn. Tím je dosažena, především při velkém výskytu menších ploch, značná redukce segmentů a zároveň nedochází k nežádoucímu vypouštění osamocených ploch.

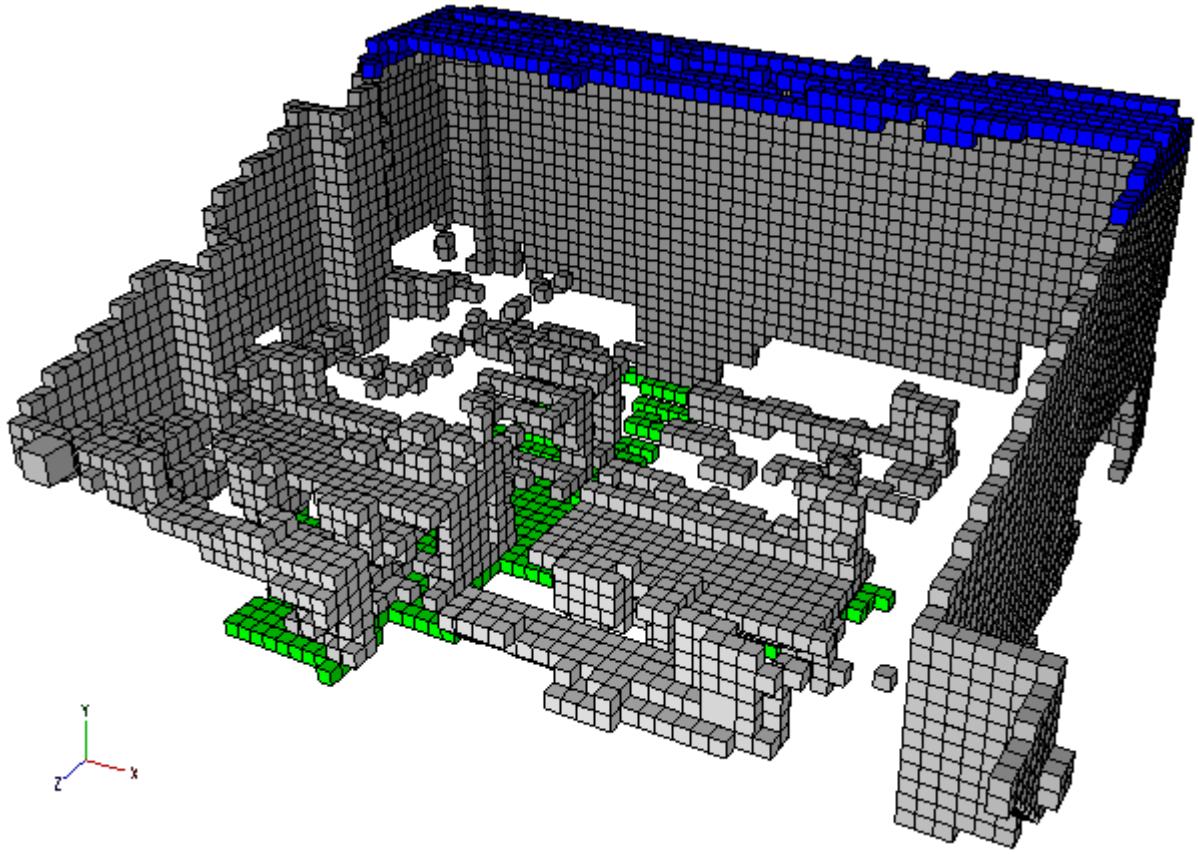
Na obrázku níže je opět znázorněna aplikace této metody na testovací příklad. V tomto případě došlo k segmentaci 137 objektů. Bez redukce menších segmentů by byl jejich výsledný počet přibližně dvojnásobný.



Obr. 22.: Segmentace objektů aplikovaná na detekované plochy

5.5 Sémantika

Při vytváření map pro účely navigace robotu je žádoucí zredukovat počet detekovaných objektů na minimum tak, aby byla výsledná mapa co možná nejjednoduší. Z tohoto důvodu a také pro možnost snazšího převádění 3D mapy do její 2D reprezentace byla implementována metoda jednoduché *sémantiky*. Metoda umožňuje definovat dvě horizontální pásma – zde označovaná jako *podlaha* a *strop*. Každé pásmo je definováno svojí vzdáleností od místa skenování a svým rozsahem. Pásmo může být buďto obarveno (zelená pro podlahu, modrá pro strop), nebo může být uplně odstraněno. Takto lze z dalšího procesu mapování a posléze navigace robotu vypustit ty segmenty, které díky své poloze nemohou ovlivnit robot při jeho pohybu a ušetřit tak výpočetní čas například při hledání cesty.



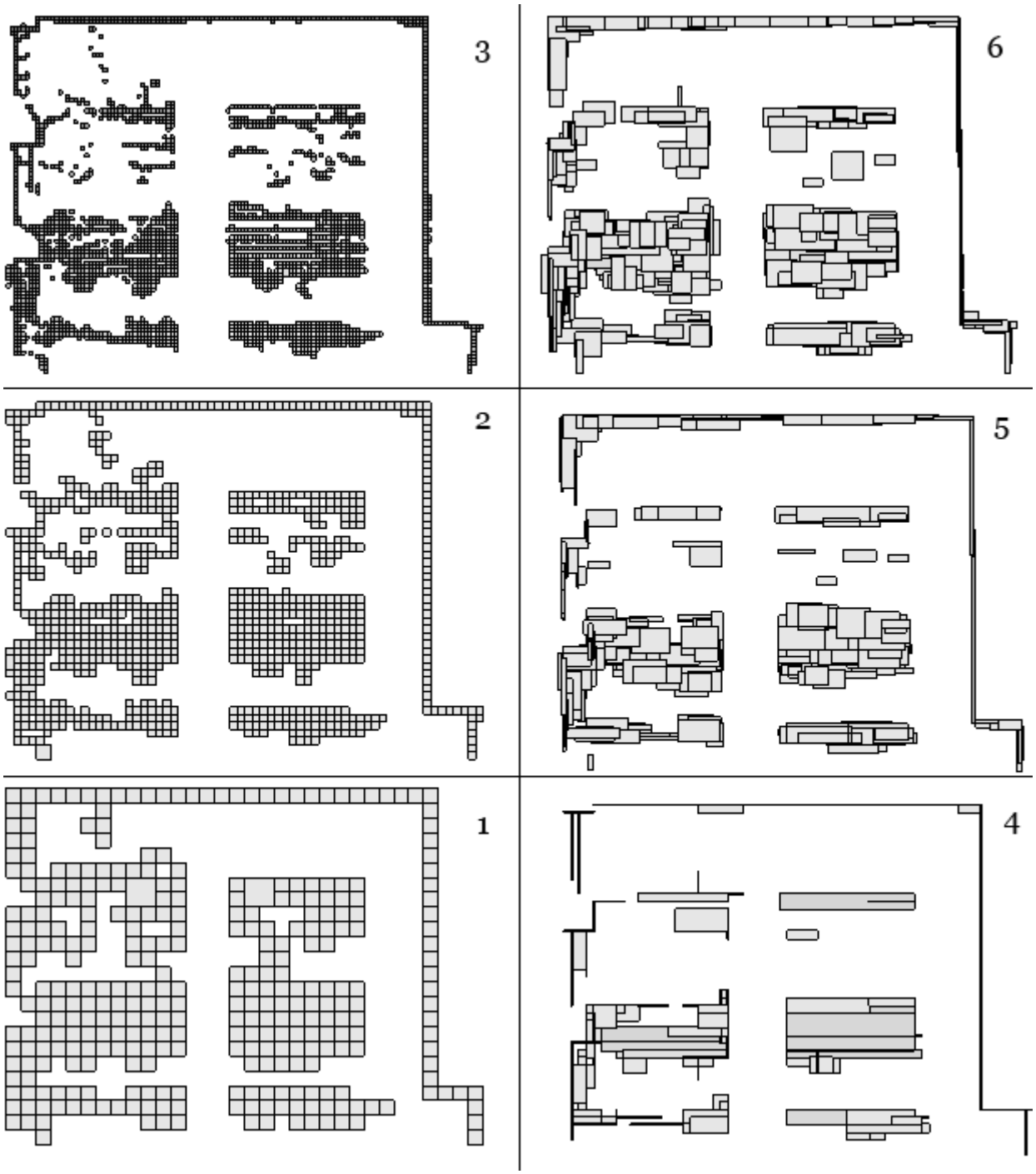
Obr. 23.: Obarvení podlahy a stropu

5.6 Porovnání metod

V předcházejících odstavcích byly prezentovány dvě metody zpracování mračna bodů, které vedou k vytvoření mapy okolí vhodné pro využití mobilním robotem. Ačkoliv se jedná o dosti odlišné přístupy lze je porvnat z několika hledisek jako je výpočetní doba a počet výsledných segmentů. Pro potřeby porovnání bylo provedeno šest různých způsobů segmentace. Pro každou metodu vždy tři příklady s různě nastavenými parametry algoritmů, jejich popis je obsažen v následující tabulce a znázorněné výsledky na Obr. 24. Pro úplnost uvádím procesor Intel Celeron M 1.4 GHz, na kterém byly výpočty provedeny.

Př. č.	Typ metody	Poznámka	Počet segmentů	Doba výpočtu
1	Oktalový strom	Hloubka stromu 5	1 148	0,55 s
2	Oktalový strom	Hloubka stromu 6	4 841	0,57 s
3	Oktalový strom	Hloubka stromu 7	11 885	0,76 s
4	Detekce ploch	Předzpracování Oktalovým stromem hloubky 6	111	1,48 s
5	Detekce ploch	Předzpracování konstantním rozestupem bodů	462	2,31 s
6	Detekce ploch	Bez přezpracování	595	16,88 s

Tab. 10.: Srovnání metod segmentace



Obr. 24.: Výsledná reprezentace mapy jednotlivých způsobů segmentace při pohledu shora.

Z tabulky jsou na první pohled patrné klady a zápory jednotlivých metod. Oktalový strom je velice rychlý na výpočet. I při vyšší hloubce stromu se doba výpočtu zvýšila jen nepatrně, přibližně o 200 ms. Počet generovaných segmentů v hloubce stromu 5, která je vzhledem k velikosti mřížky na hranici použitelnosti, je 1148.

Segmentace pomocí detekce ploch je oproti oktalovému stromu výpočtově náročnější. Doba výpočtu se odvíjí především od kvality a množství detekovaných úseček. Při předzpracování dat oktalovým stromem je doba výpočtu 1,48 s přibližně trojnásobná. Bez předzpracování však může být množství detekovaných úseček v řádu tisíců a doba

výpočtu tak výrazně narůstá do desítek sekund až několika minut. Nespornou výhodou této metody však je počet generovaných segmentů, jenž je přibližně desetkrát menší než u oktalového stromu.

Obě metody se liší ještě v jedné vlastnosti, ve které naopak vítězí oktalový strom. Jde o to, že u oktalového stromu jsou ve výsledku zpracovány naprosto všechny vstupní body a proto zde nehrozí výpadky v detekci malých objektů. To je nepříjemnou vlastností segmentace pomocí ploch, jelikož v místech kde je malý počet vstupních bodů nemusí vůbec dojít k detekci úsečky a potažmo plochy a segmentu.

5.7 Související práce

Na závěr této kapitoly bych ještě rád uvedl dvě velice silné metody vyskytující se v literatuře, které mohou sloužit jako inspirace pro případné vylepšení mnou implementovaných algoritmů.

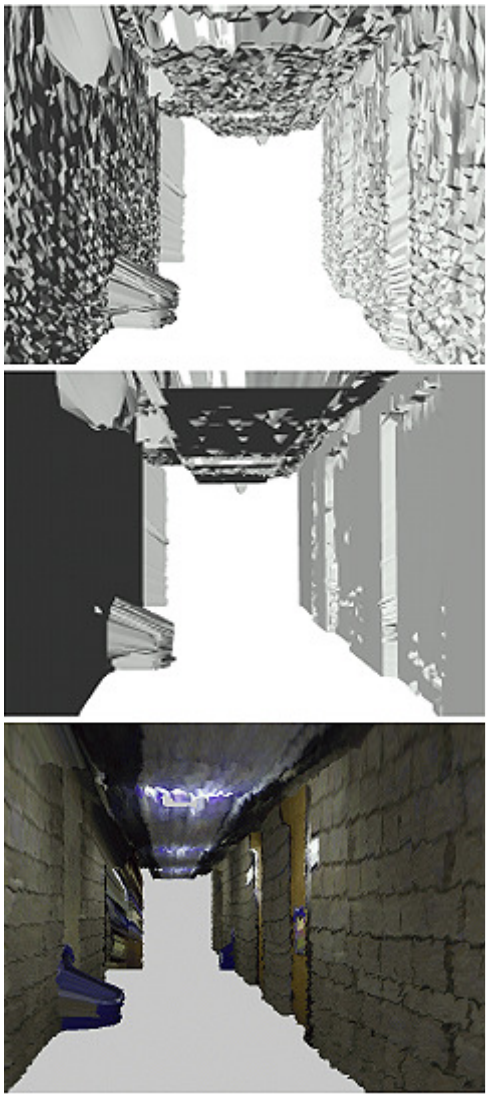
Prvním příkladem jsou práce S.Thruna [5] [7] [8] a jedná se o praktické rozšíření algortimu popsaného v odstavci 5.2. Předmětem je využití autonomního robotu pro tvorbu strukturálních map budov a experimentální mapování opuštěných dolů [6].

Robot znázorněný na Obr. 25 je vybaven dvěma laserovými skenery. Jeden skener je namířen dopředu a slouží pro 2D mapování a navigaci. Druhý skener je připevněn ve vertikální poloze tak, že rovina měření je kolmá na první skener a také na směr pohybu robotu. Tento dálkoměr při pohybu robota skenuje příčné průřezy budovy.

Robot je také vybaven panoramickou kamerou. Panoramická kamera využívá malého zrcadla, který poskytuje obraz oblasti pokrytý vertikálním skenerem. Toto vylepšení umožňuje spojení informace o vzdálenosti a barevné informace v rovině měření vertikálního skeneru.

Obr. 25.: Vlevo: Robot požitý v práci [6]. Vpravo detail zrcadla panoramické kamery

Textury jsou získány z panoramické kamery, podél proužku zobrazeného v pravé části obr. 25, které korespondují z měřením získaného z vertikálně usazeného dálkoměru. Tyto pruhy jsou zaznamenávány v pravidelných intervalech a následně složeny dohromady v hrubou texturu. Textury jsou posléze namapovány na extrahované planární plochy.



Obr. 26.: Postup mapování popsaný v práci S. Thruna [5]

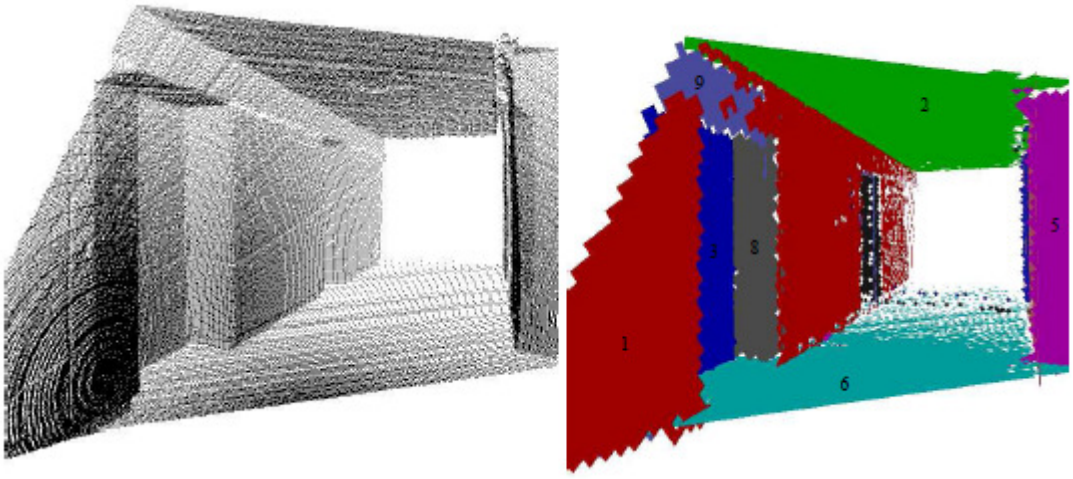
Obrázek vlevo ukazuje postup celého procesu mapování. V horní části jsou vidět hrubá data měření, z nichž jsou vytvořeny plochy způsobem popsaným v 5.2. Je dosti patrné, že data jsou zatížena značnou chybou způsobenou především nedokonalou odometrií robota a systematickou chybou měření.

Obrázek uprostřed představuje plochy zpracované EM (Expectation Maximization) pravděpodobnostním algoritmem. Tento algoritmus využívá předpokladu, že interiéry budov obvykle obsahují množství velkých, dobře definovaných ploch. Celý princip algoritmu je dosti složitý, ale velmi zjednodušeně se dá popsat jako hledání nejlepší možné aproximace co největšího množství malých polygonů jedinou plochou. Algoritmus může být sám o sobě výpočtově velmi náročný (až několik hodin), ale díky implementaci Lagrangovych multiplikátorů ho lze použít v reálném čase. Aplikací této metody na hrubá data dojde k nejméně desetinásobné redukci ploch a jejich výraznému zjednodušení.

Na posledním obrázku lze již vidět výsledek celého postupu, kde je na zjednodušenou reprezentaci vstupních dat namapována textura z panoramické kamery. Panoramickou kameru pro získávání textur lze nahradit také například jedním nebo více vhodně umístěnými fotoaparáty.

Druhým příkladem je práce H. Surmanna [4], která je pokračováním předcházejících výzkumů [1] [2]. Je zde prezentována velice silná metoda extrahování ploch vycházející ze stejného principu jako metoda v práci S. Thruna. Jedná se o kombinaci algoritmu ICP (iterative closest point) a algoritmu RANSAC (random sample consensu).

Algoritmus RANSAC slouží k prokládání velkého množství dat množinou definovaných geometrických primitiv (v tomto případě ploch). Nejprve se z mračna bodů vybere náhodně *N* prvků a pro které se stanoví parametry plochy. V dalším kroku se testuje dané množství dalších bodů, zda do této plochy náleží v dané toleranci. Pokud je nalezeno dostatečné množství takovýchto bodů je vytvořena plocha, jinak algoritmus začíná s hledáním od znova s jinými nahodně vybranými daty. Vytvořené plochy však nejsou ohraničeny. Pro definování velikosti plochy, která náleží jejím bodům je použita metoda kvadratického stromu (obdoba oktalového stromu pro 2D). ICP algoritmus je zde použit pro spojování více provedených skenů do jedné množiny. Příklad aplikace této metody je na následujícím obrázku.

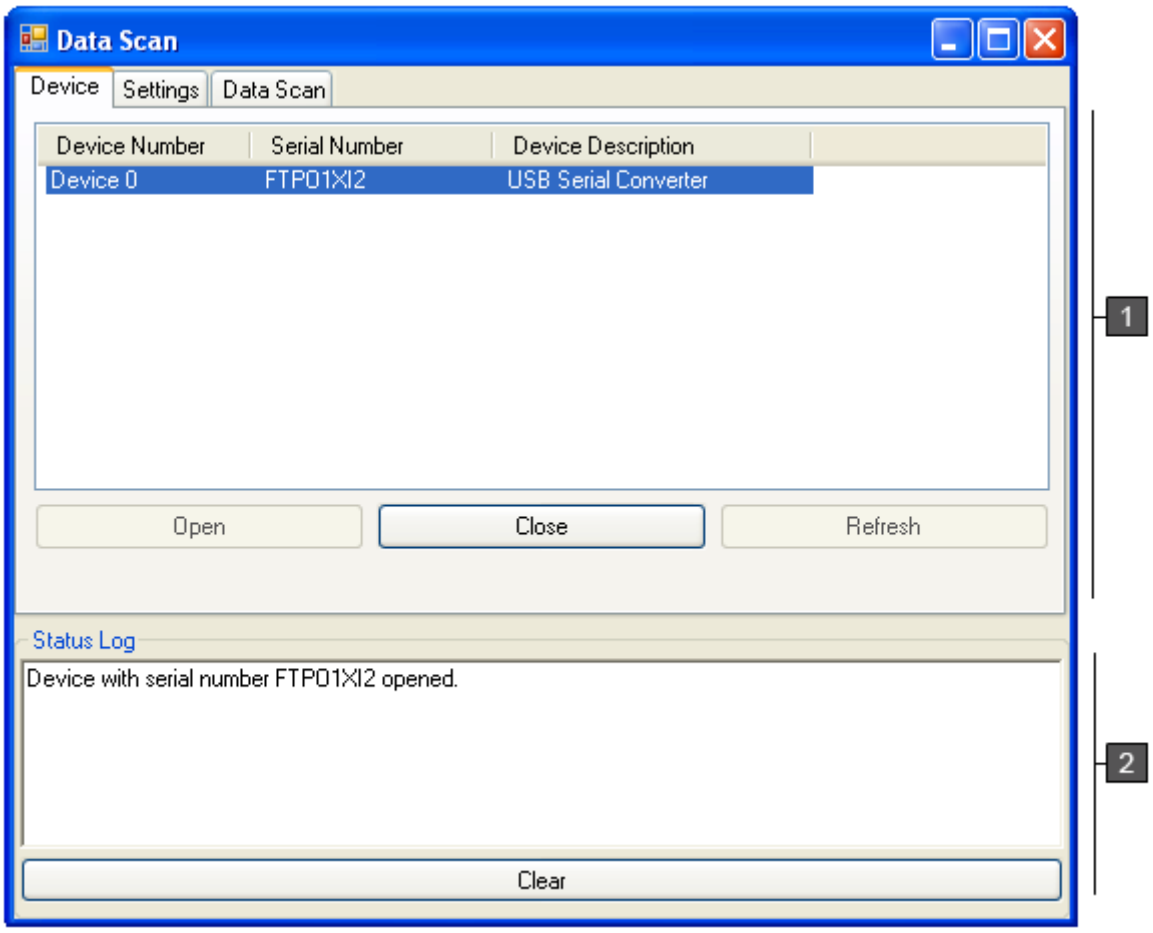


Obr. 27.: Extrakce ploch pomocí metody RANSAC [4]

6 POPIS APLIKACE PRO ČTENÍ DAT

6.1 Formulář navázání komunikace

Formulář aplikace pro čtení dat je rozdělen na dva horizontální bloky. Část označená číslem 1 obsahuje záložky *Device* – navázání komunikace, *Settings* – nastavení parametrů a *Data Scan* – provedení měření. Část označená číslem 2 je tzv. *status* okno, kde se vypisují všechny zprávy o úspěšně a neúspěšně provedených akcí.

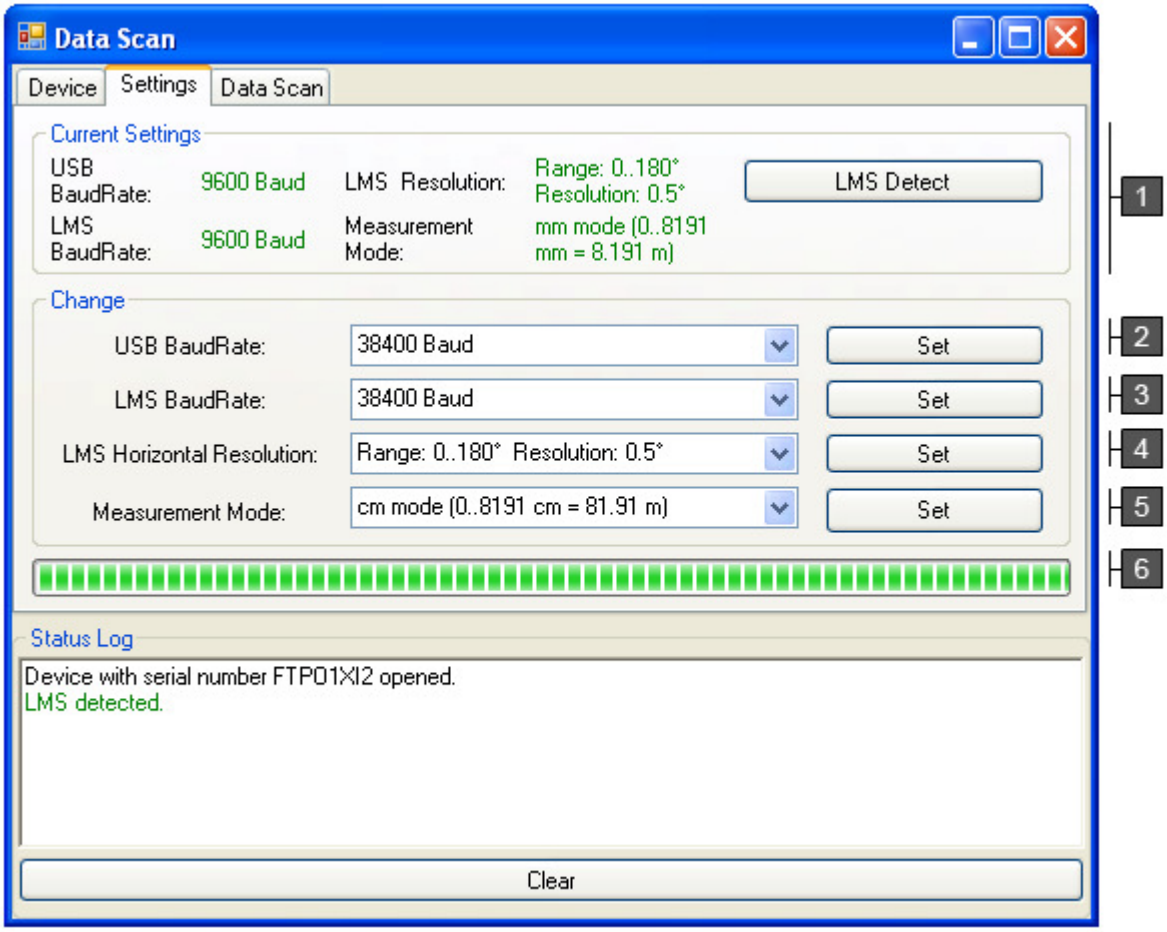


Obr. 28.: Formulář aplikace pro čtení dat

Záložka *Device* obsahu seznam USB převodníků připojených k počítači. Pro každou jednotku je vypsáno její pořadové číslo, sériové číslo, a její popis. Pro otevření komunikační linky, je potřeba vybrat patřičnou jednotku ze seznamu a stisknout tlačítko *Open*. Uzavřít linku lze pomocí tlačítka *Close*. Tlačítko *Refresh* slouží k znovunačtění připojených jednotek. Jakmile je vypsána zpráva o úspešnosti otevření komunikační linky, lze pokračovat nastavováním skeneru.

6.2 Formulář nastavení

Pro nastavování všech potřebných parametrů skeneru je určena záložka Settings. Blok na obrázku označený číslem 1 udává aktuálně nastavené hodnoty. Po připojení skeneru je nutné načíst jeho nastavení pomocí tlačítka *LMS Detect*. Úspěšné přijetí *status telegramu* je doprovázeno vypsáním aktuálních hodnot a potvrzením zprávou v *status* okně. Pokud dojde k vypsání chybové hlášky je patrně nastavena jiná přenosová rychlost USB převodníku (2) a LMS a je nutné rychlost převodníku přenastavit, nebo resetovat LMS, které má vždy po spuštění nastavenou přenosovou rychlost na 9 600 Bd.



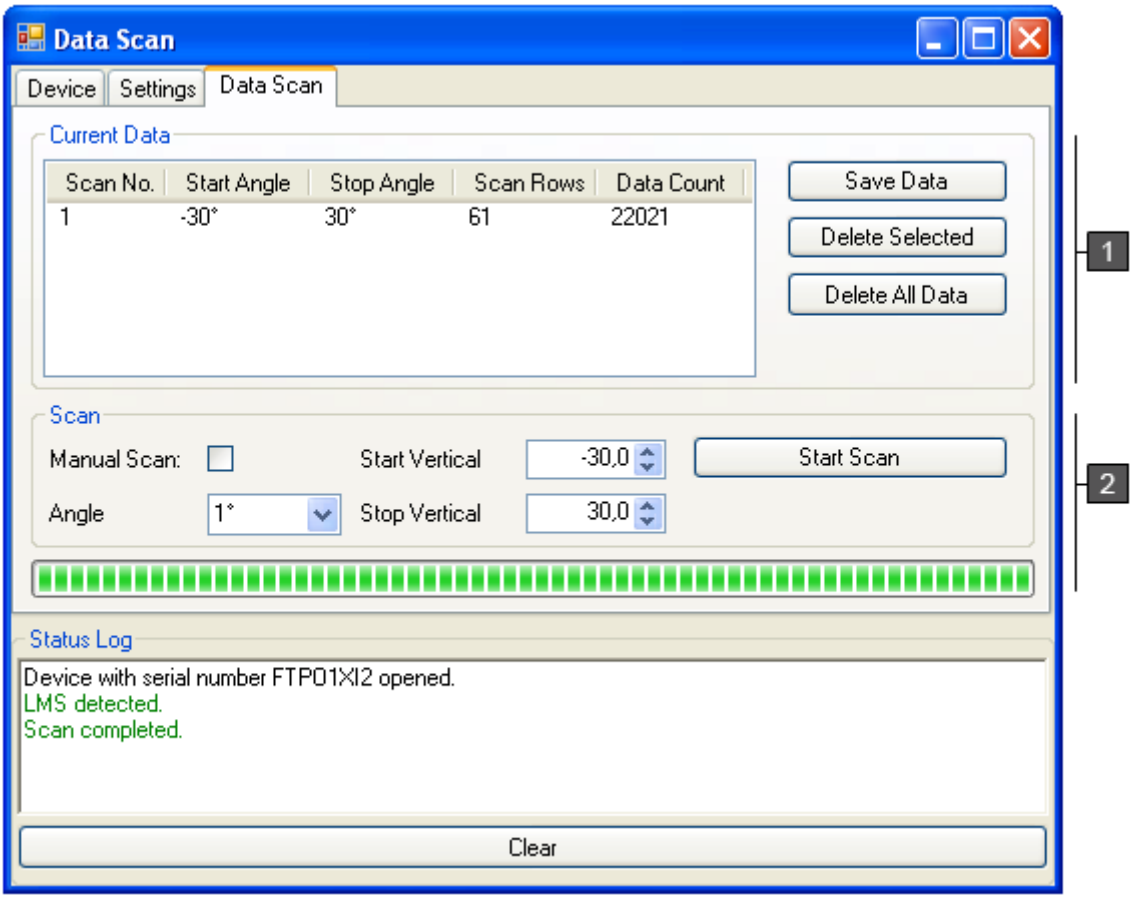
Obr. 29.: Záložka Settings

Jakmile jsou načteny aktuální hodnot, lze přejít k jejich přenastavení. Změnu přenosové rychlosti LMS lze provést vybráním příslušné hodnoty v řádku 3 a stiskem tlačítka *Set*. Je nutné mít na paměti, že po každé změně rychlosti LMS je nutné změnit také rychlost USB převodníku na stejnou hodnotu, jinak nebude možné dále se skenerem komunikovat. Obdobným způsobem se nastavuje úhlové rozlišení skeneru v řádku 4 a jednotek měření v řádku 5.

Průběh každé akce je signalizován *progress barem* označeným číslem 6. Přenastavení rychlosti a úhlového rozlišení by mělo proběhnout prakticky okamžitě. Změna měřených jednotek si však vyžádá přibližně 8 sekund. Pokud dojde během změny nastavení k vypsání chybové hlášky tak se nastavení hodnoty nezdařilo a je nutné tuto akci opakovat. Po nastavení všek patřičných hodnot lze přejít na záložku měření – *Data Scan*.

6.3 Formulář měření

Záložka pro měření – *Data Scan* je rozdělena na blok 1 pro správu naměřených dat a blok 2 pro provedení měření. Blok 2 umožňuje nastavit počáteční a koncový vertikální úhel skenování a krok, po kterém se má měření provádět. Po nastavení patřičných úhlů a kroku lze měření spustit tlačítkem *Start Scan*. Pokud chce uživatel provést pouze měření jedné roviny lze tak učinit zaškrtnutím položky *Manual Scan*. Průběh aktuálního měření je opět signalizován *progress barem*.



Obr. 30.: Záložka Data Scan

Po úspešném měření dojde v seznamu bloku 1 k vypsání údajů o tomto měření. Je zde uvedeno pořadové číslo měření, počáteční a koncový úhel, počet naskenovaných rovin a celkový počet všech naměřených hodnot. Všechna data obsažené v seznamu lze uložit do textového souboru tlačítkem *Save Data*, kdy dojde k otevření klasického dialogového okna pro ukládání na disk. Jednotliná měření lze odstranit jejich označením v seznamu a stiskem tlačítka *Delete Selected*. Všechna měření lze odstranit tlačítkem *Delete All Data*.

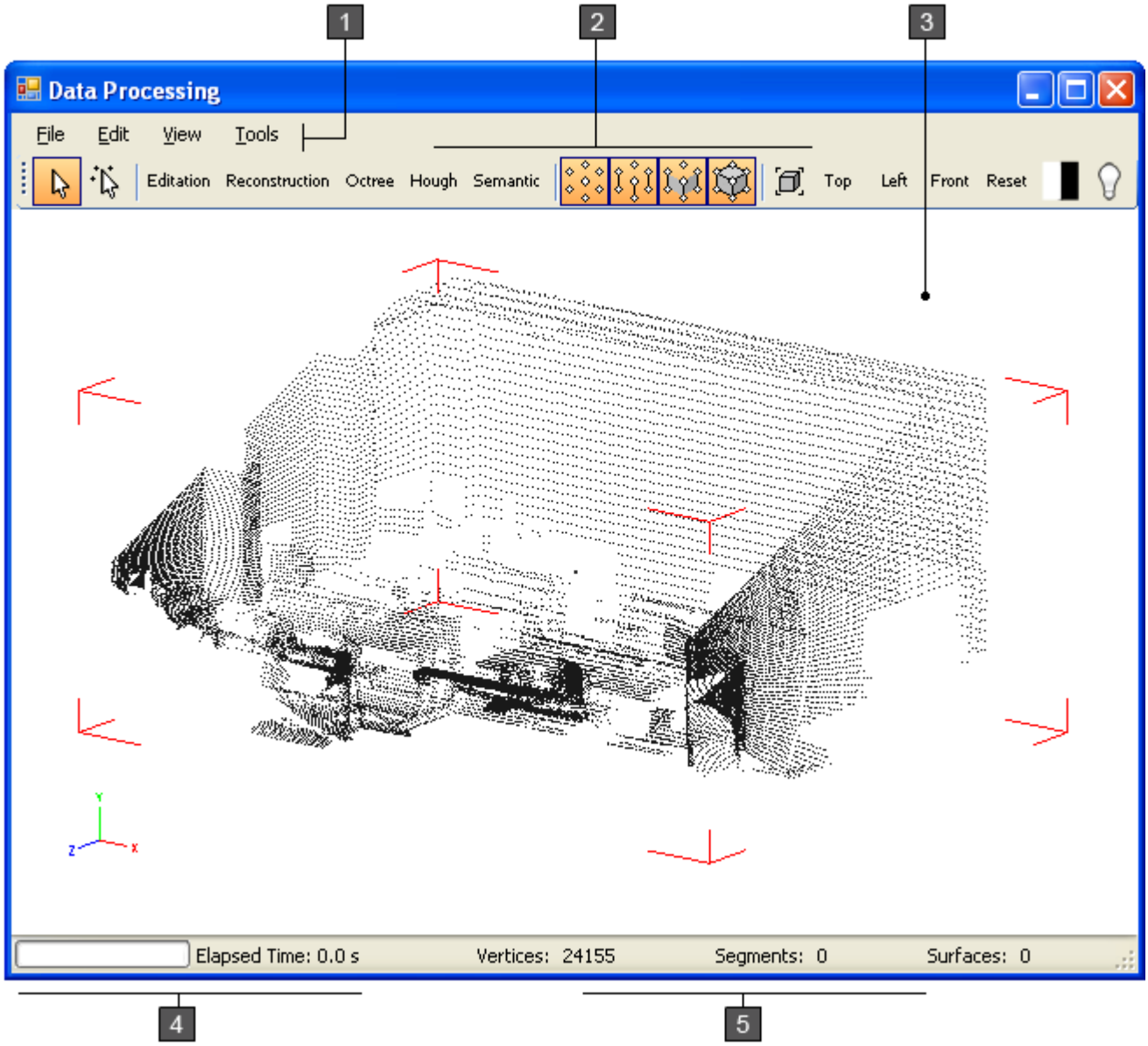
6.4 Minimální požadavky

Pro správnou funkci aplikace je nutné zajistit tyto minimální požadavky:

- OS Windows Server 2003, Windows Vista; Windows XP
- Nainstalovaný .Net Framework 2.0 nebo vyšší
- Procesor 400 MHz (doporučeno 1GHz)
- Pamět 96 MB RAM (doporučeno 256 MB)
- Nainstalované ovladače FTDI (přiloženy na CD)

7 POPIS APLIKACE PRO ZPRACOVÁNÍ DAT

Hlavní okno aplikace obsahuje klasické Menu (1) všech dostupných funkcí programu, panel nástrojů (2) pro rychlý přístup k nejčastěji používaným funkcím, plochu, ve které probíhá veškerá vizualizace (3) a ve spodní části pak *progress bar* (4) signalizující průběhy výpočtů a údaje o době výpočtu a počtu bodů, ploch a segmentů objektu (5).



Obr. 31.: Hlavní okno aplikace

Ovládání probíhá pomocí myši a klávesnice. Levé tlačítko slouží k výběru jednotlivých mračen bodů nebo samotných bodů. Posun pohledu je umožněn stlačením kolečka myši, rotace stlačením kolečka myši spolu se stlačenou klávesou Alt, a přibližování a oddalování pohledu rotací kolečka myši.

7.1 Menu a panel nástrojů

Hlavní menu obsahuje všechny funkce programu a přístup k nástrojům. Všechny funkce mají přiřazeny klávesové zkratky pro snazší obsluhu. Následuje stručný výčet prvků menu.

Položka *File* obsahuje klasické funkce pro otevření a uložení souboru a funkci pro import souboru uložený aplikací pro čtení dat. *Edit* obsahuje funkce *Undo* a *Redo*, kde je možný vždy pouze jeden krok vpřed či vzad. *View* obsahuje funkce pro ovládání zobrazení jako přepínání mezi orthografickou a perspektivní projekcí, přednastavené úhly pohledu, možnost invertovat barvy zobrazení a aktivaci a deaktivaci mřížky, souřadných os atp. Položka *Tools* poskytuje přístup k formulářům popsaným dále.

Panel nástrojů obsahuje nejčastěji používané funkce, pro snazší orientaci jsou opatřeny popisem.

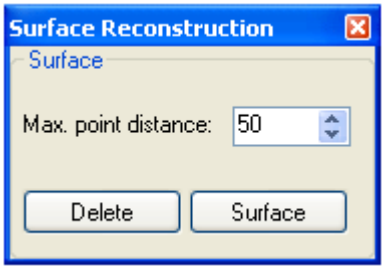
Obr. 32.: Panel nástrojů

V levé části jsou nástroje pro výběr a to celých mračen bodů (1), nebo jednotlivých bodů (2). Výběr jednotlivých bodů se uskutečňuje tažením myší a následně je lze odstranit (klávesa Del). Dalšími jsou tlačítka pro rychlý přístup k formulářům (3). Následuje možnost zapínat a vypínat zobrazení jednotlivých typů elementů (4). Poslední částí (5) jsou prvky pro ovládání zobrazení.

7.2 Formulář editace

Formulář editace poskytuje možnost translace (část *Position*) a rotace (část *Rotation*) zvoleného objektu vzhledem ke globálnímu souřadnicovému systému. Tlačítko *Align With Grid* zarovná objekt na mřížku. V části *Editation* jsou tlačítka pro odstranění vygenerovaných čar, ploch, segmentů nebo celého objektu.

7.3 Formulář rekonstrukce povrchů

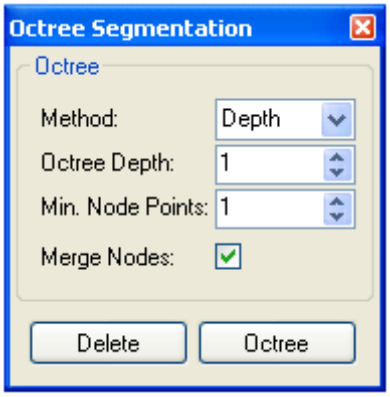


Obr. 33.: Formulář rekosntrukce ploch

Formulář rekonstrukce povrchů je imlepmentací metody popsané v kapitole 5.2.

Jediným parametrem je zde maximální vzdálenost v centimetrech mezi sousedními body, které mají vytvořit plochu. Po zadání vhodné hodnoty se tlačítkem *Surface* provede výpočet. Vzniklé plochy lze odstranit tlačítkem *Delete.*

7.4 Formulář oktalového stromu



Obr. 34.: Formulář Oktalového stromu

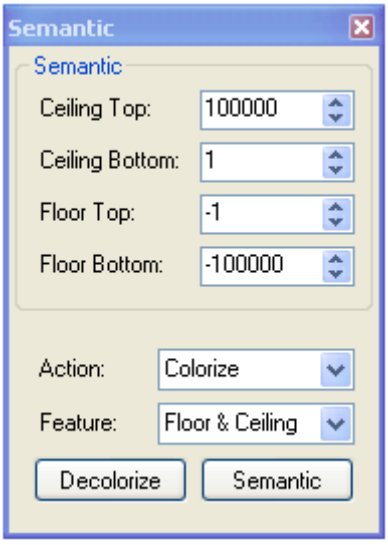
Formulář oktalového stromu obsluhu implemetaci metody z kap. 5.3.

První položkou *Method* je volba metody podle, které se bude oktalový strom konstruovat. Na výběr je konstrukce podle zadané hloubky stromu (*Depth*), nebo podle maximální velikosti hrany vygenerovaného segmentu v cm (*Node Size*).

Druhá položka slouží k zadání výše zmíněného parametru v závisloti na zvolené metodě.

Třetí položka *Min. Node Points* udává, kolik nejméně bodů má obsahovat list stromu. Pokud tento list bude obsahovat méně než zadaná hodnota, bude z grafu vypuštěn. Zaškrtnutím *Merge Nodes* se určuje, zda se mají uzly v předposlední úrovni stromu dělit na své oktany, pokud jich vznikne všech osm. Jinak řečeno osm listů výsledného stromu, které mají stejného předchůdce lze nahradit tímto předchůdcem. Tím dojde k redukci výsledného počtu segmentů. Tlačítkem *Octree* se provede konstrukce stromu, tlačítkem *Delete* ho lze odstranit.

7.5 Formulář sémantiky



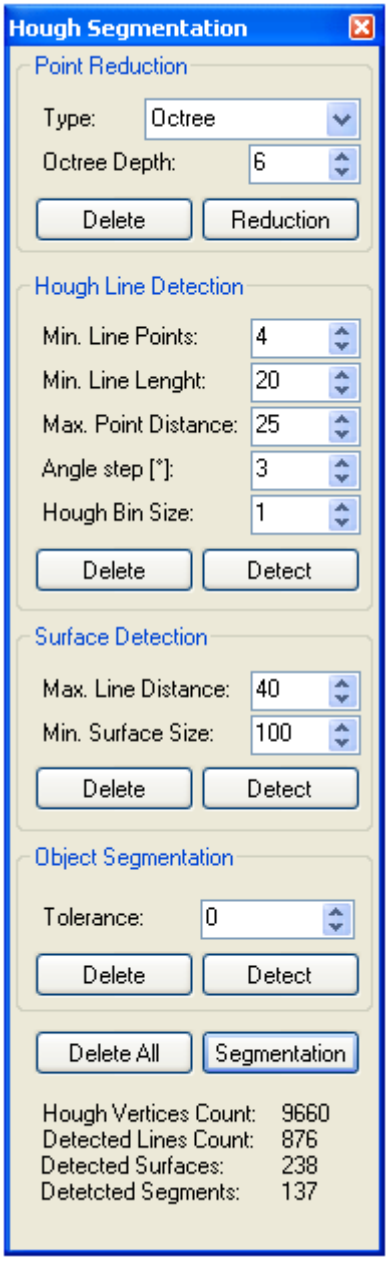
Obr. 35.: Formulář sémantiky

Formulář sémantiky umožňuje obarvovat nebo úplně odstranit dvě horizontální pásma tak, jak je popsáno v kap. 5.5.

Pro obě pásma *podlahy (Floor)* a *stropu (Ceiling)* se zadává vzdálenost jejich spodní a horní plochy od místa uložení skeneru. Např. pokud byl skener při měření ve výšce 150 cm, podlaha bude patrně začínat ve vzdálenosti -150 cm. Je ale nutno počítat také s výškou generovaných segmentů, tedy pro segmenty výšky 20 cm, bude zmíněná hodnota přibližně -140 cm.

Druhá část nabízí volbu akce, zda se mají segmenty pouze obarvit nebo odstranit a volbu, zda se má akce provést pouze na pásmo podlahy, stropu nebo na obojí. Akce se provede tlačítkem *Semantic*, tlačítkem *Decolorize* lze segmenty odbarvit.

7.6 Formulář segmentace pomocí Houghovy transformace



Obr. 36.: Formulář segmentace

Formulář je dělen na po sobě následující časti tak, jak byly popsany v kap. 5.4. Každou akci tak lze provést samostatně příslušným tlačítkem nebo ji odstranit tlačítkem Delete. Všechny akce algoritmu podle zadaných parametru lze provést najednou, pomocí tlačítka *Segmentation* v dolní části formuláře.

První část *Point Reduction* představuje možnost předzpracování mračna bodů. Možnost volby je z oktalového stromu a poměrných vzdáleností mezi body. Parametr pro strom je jeho hloubka nebo pro druhou metodu vzdálenost v cm.

Druhá část obsluhuje detekci úseček pomocí Houghovi transformace. Prvním parametrem je minimální počet bodů, které mají tvořit úsečku. Druhý parametr udává minimální délku úsečky v cm. Třetí parametr je maximální vzdálenost mezi dvěma sousedními body na úsečce. Uhlový krok (*Angle step*) udává, po jakém kroku se mají testovat přímky v Houghově prostoru, jinak řečeno na kolik sloupců se má rozdělit Houghův prostor. Hodnota 1 znamená 180 testů, hodnota 3 znamená 60 testů atd. Tedy čím větší hodnota, tím rychlejší výpočet. *Hough Bin Size* je velikost jednoho pole Houghova zásobníku, jinak řečeno na kolik řádků se má rozdělit Houghův prostor. Vyšší hodnota znamená rychlejší výpočet a vetší pravděpodobnost nalezení úsečky. Pro předzpracovaná data pomocí oktalového stromu však obvykle plně dostačuje hodnota nastavená na 1.

Část *Surface Detection* slouží ke generování ploch z nalezených úseček. Prvním parametrem je maximální vzdálenost koncových bodů dvojice úseček, které mají vytvořit plochu a druhým parametrem je minimální obsah plochy.

Část Object Segmentation obsluhuje segmentaci objektů z ploch. Parametr *Tolerance* udává toleranci podle, které mají být menší objekty pohlceny většími.

Dolní část formuláře obsahuje počty jednotlivých generovaných elementů.

7.7 Minimální požadavky

Pro správnou funkci aplikace je nutné zajistit tyto minimální požadavky:

- OS Windows Server 2003, Windows Vista; Windows XP
- Nainstalovaný .Net Framework 2.0 nebo vyšší
- Procesor 800 MHz (doporučeno 1.8 GHz)
- Pamět 96 MB RAM (doporučeno 256 MB) a grafická karta s podporou OpenGL

8 ZÁVĚR

Cílem této práce bylo využití 2D laserového vertikálně polohovatelného dálkoměru pro potřeby 3D mapování prostředí autonomních mobilních sytémů. 3D mapování okolního prostředí je stále ne zcela prozkoumanou a vyvíjející se oblastí mobilní robotiky.

V kapitolách 2. a 3. byly představeny použité automatizační prostředky, zejména laserový dálkoměr SICK LMS 291 s jeho polohovatelnou konstrukcí a programové prostředky v podobě platformy .Net, jazyka C# a technologie OpenGL.

Čtvrtá kapitola se zabývala navrženým řešením pro provedení měření a čtení dat z dálkoměru. Byl zde popsán postup komunikace s dálkoměrem, nastavování jeho parametrů pomocí specifických telegramů a princip měření. Výsledkem je samostatná aplikace umožňující komunikaci s dálkoměrem a následné skenování okolí v různých úhlových rozsazích a zaznamenání naměřených dat v podobě mračna bodů.

Pátá kapitola představuje zpracování naměřených dat pomocí speciálních algortimů mapování prostředí. Konkrétně byly implementovány metody rekonstrukce povrchu, oktalového stromu, a segmentace objektů pomocí Houghovi transformace. Rekonstrukce povrchu jako metoda reprezentace okolí je vhodná spíše pro potřeby člověka. Oktalový strom a segmentaci objektů pomocí Houghovi transformace lze využít k 3D mapování okolí, vhodného pro následnou navigaci a lokalizaci autonomních mobilních systémů.

Implementované algoritmy spolu s 3D laserovým dálkoměrem poskytují dobrý základ pro senzorickou soustavu mobilního robotu. Je však nutné brát v potaz, že se jedná pouze o odrazový můstek v celé etapě konstrukce plně autonomních systémů, který musí být doplněn nejen o kvalitní odometrii, *scan-matching*, a SLAM algoritmy.

SEZNAM POUŽITÉ LITERATURY

[1] SURMANN, H.; LINGEMANN, K.; NUCHTER, A.; HERTZBERG, J. A 3D laser range finder for autonomous mobile robots, in: *Proceedings of the 32nd International Symposium on Robotics (ISR'01), Seoul, Korea, April 2001*, pp. 153–158.

[2] SURMANN, H.; LINGEMANN, K.; NUCHTER, A.; HERTZBERG, J. Fast acquiring and analysis of three-dimensional laser range data, in: *Proceedings of the Sixth International Fall Workshop Vision, Modeling, and Visualization (VMV'01), Stuttgart, Germany, November 2001*, pp. 59–66.

[3] SURMANN, H.; NUCHTER, A.; HERTZBERG, J. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments, in: *Robotics and Autonomous Systems 45, 2003, pp. 181–198*.

[4] SURMANN, H.; NUCHTER, A.; HERTZBERG, J. Automatic Model Refinement for 3D Reconstruction with Mobile Robots. 2004

[5] THRUN, S; MARTIN, C.; LIU, Y.; HAHNEL, D.; EMERY-MONTEMERLO, R.; CHARKABARTI, D.; BURGARD, W. A real-time expectation maximization algorithm for acquiring multi-planar maps of indoor environments with mobile robots. IEEE Transactions on Robotics, 2004, pp. 433-443.

[6] THRUN, S.; THAYER, S.; WHITTAKER, W.; BAKER, C.; BURGARD, W.; FERGUSON, D.; HÄHNEL, D.; MONTEMERLO, M.; MORRIS, A.; OMOHUNDRO, Z.; REVERTE, C.; WHITTAKER., W. Autonomous exploration and mapping of abandoned mines. IEEE Robotics and Automation, 2005.

[7] THRUN S. Robotic mapping: A survey. In G. Lakemeyer and B. Nebel, editors, Exploring Artificial Intelligence in the New Millenium. Morgan Kaufmann, 2002.

[8] THRUN Sebastien; FOX Dieter; BURGARD Wolfram. A realtime algorithm for mobile robot mapping with applications to multi-robot and 3d mapping, in *IEEE International Conference on Robotics and Automation, San Francisco, 2000, pp. 321–328*.

[9] SURMANN, H.; LINGEMANN, K.; NUCHTER, A.; HERTZBERG, J. Accurate Object Localization in 3D Laser Range Scans. 2005.

[10] WULF Oliver; WAGNER Bernardo. <wulf@rts.uni-hannover.de> *Fast 3D scanning methods for laser measurement systems*. Institute for Systems Engineering, University of Hannover, Germany Email: {wulf,wagner}@rts.uni-hannover.de

[11] SIDDHARTH Jain. A survey of Laser Range Fading. December 2003.

[12] SICK LMS Technical Description. SICK AG Germany, 2003. Dostupné z: http://www.sick-automation.ru/images/File/pdf/LMS%20Technical%20Description.pdf

[13] Telegrams for Configuring and Operating the LMS2xx Laser Measurement Systems. SICK AG Germany, 2006. Dostupné z: http://www.mysick.com/saqqara/pdf.aspx?id=im0015003

[14] SHARP, John. *Microsoft Visual C# 2005 : Krok za krokem*. 1. vyd. Brno: Computer Press, 2006. 528 s. ISBN 80-251-1156-3.

[15] Software Application Development D2XX Programmer's Guide. Future Technology Devices International Ltd. 2008. Dostupné z: http://www.ftdichip.com/Documents/ProgramGuides/D2XX_Programmer%27s_Guide(FT_000071).pdf

[16] FT232R USB UART IC Datasheet. Future Technology Devices International Ltd. Version 2.01, 2005. Dostupné z: http://www.ftdichip.com/Documents/DataSheets/DS_FT232R.pdf

[17] HORKEL M.; KÁKONA J. Převodník USB na RS232. 2008. Dostupné z: http://www.mlab.cz/Modules/CommSerial/USB232R01B/DOC/USB232R01B.cs.pdf

[18] PŘISPĚVATELÉ WIKIPEDIE. *Hough Transform - Wikipedia* [online]. [cit. 29.4.2009]. Dostupné z: http://en.wikipedia.org/wiki/Hough_transform

[19] PUŠ, Petr. Poznáváme C# a Microsoft .NET. [online]. [cit. 18.4.2009]. Dostupné z: http://www.zive.cz/Clanky/Poznavame-C-a-Microsoft-NET--1dil/sc-3-a-120978/

[20] TIŠNOVSKÝ, Pavel. Grafická knihovna OpenGL. [online]. [cit. 29.4.2009]. Dostupné z: http://www.root.cz/clanky/graficka-knihovna-opengl-1/

[21] UHER, Ondřej. 3D laserový dálkoměr vertikálně polohovatelný. Brno, 2008. 37 s. Bakalářská práce na fakultě strojního inženýrství VUT na Ústavu automatizace a informatiky. Vedoucí práce Ing. Tomáš Marada, Ph.D.

PŘÍLOHY

Přiložené CD obsahuje:

- Elektronickou podobu této diplomové práce (Text/DP_Fritz.pdf)
- Zdrojový kód vytvořených aplikací (Aplikace_zdrojove_kody/)
- Zkompilované aplikace (Aplikace_zkompilovano/)
- Ukázku naměřených dat (Data_mereni/)
- Ovladače FTDI (FTDI_ovladace/)