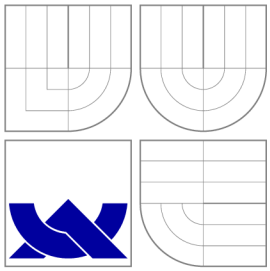
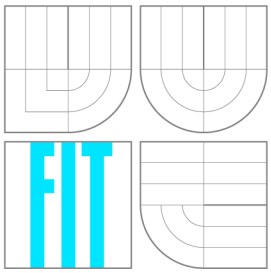


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE CESTY PRO AUTONOMNÍ VOZIDLO

ROAD DETECTION FOR AUTONOMOUS CAR

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MATÚŠ KOMORA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL ŠPANĚL, Ph.D.

BRNO 2016

Zadání diplomové práce/15889/2015/xkomor00

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2015/2016

Zadání diplomové práce

Řešitel: **Komora Matúš, Bc.**
Obor: Počítačová grafika a multimédia
Téma: **Detekce cesty pro autonomní vozidlo**
Road Detection for Autonomous Car
Kategorie: Zpracování obrazu

Pokyny:

1. Seznamte se s problematikou senzorů používaných v autonomních vozidlech (kamera, Velodyne LiDAR) a základními principy zpracování získaných dat. Zaměřte se zejména na zpracování tzv. mračen bodů a zpracování obrazu ze stereo kamery.
2. Zorientujte se v současných metodách detekce cesty pro autonomní vozidla s využitím senzoru Velodyne LiDAR a kamer.
3. Vyberte vhodné metody a navrhněte rychlý detektor cesty v okolí vozidla.
4. Experimentujte s vaší implementací a případně navrhněte vlastní modifikace metod.
5. Porovnejte a diskutujte dosažené výsledky a zvažte možnosti budoucího vývoje.
6. Vytvořte stručný plakát nebo video prezentující vaši práci, její cíle a výsledky.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Splnění prvních tří bodů zadaní.

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

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

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

Vedoucí: **Španěl Michal, Ing., Ph.D.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2015
Datum odevzdání: 25. května 2016

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

L.S.

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Táto diplomová práca sa venuje problematike detekcie cesty v okolí autonómneho vozidla. Cesta sa vyhodnocuje na základe dát z laserového radaru Velodyne LiDAR. V práci je použité už existujúce riešnie, ktoré je rozšírené o strojové učenie SVM s postupným učením. Práca porovnáva staré a nové riešenie na datasete KITTI. Úspešnosť odhadu cesty je vypočítaný podľa ukazateľa F-measure.

Abstract

This thesis deals with detection of the road adjacent to an autonomous vehicle. The road is recognition is based on the Velodyne LiDAR laser radar data. An existing solution is used and extended by machine learning - a Support Vector Machine with online learning. The thesis evaluates the existing solution and the new one using a KITTI dataset. The reliability of the road recognition is then computed using F-measure.

Kľúčové slová

detekcia cesty, SVM, Lidar, strojové učenie, ROS, KITTI

Keywords

road detection, SVM, Lidar, machine learning, ROS, KITTI

Citácia

KOMORA, Matúš. *Detekce cesty pro autonomní vozidlo.* Brno, 2016. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Španěl Michal.

Detekce cesty pro autonomní vozidlo

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pana Ing. Michala Španěla Ph.D.. Uviedol som všetky literárne pramene a publikácie z ktorých som čerpal.

.....
Matúš Komora
25. mája 2016

Poďakovanie

Chcel by som poďakovať vedúcemu práce za podporu a tolerantnosť počas celej spolupráce.

Obsah

Kapitola 1

Úvod

Hľadanie cesty pre autonómne vozidlo je veľká téma posledných rokov. Automobilky sa predháňajú v tom, ktorá rýchlejšie a lepšie dokáže upozorniť šoféra vozidla pred kolíziou. Ale nie je len o bezpečnosti, ale aj o možnosti bezpečného pohybu autonómnych vozidiel medzi nami.

Táto práca sa venuje hľadaniu cesty v okolí autonómneho vozidla. Hľadanie cesty je odhadnuté z mračna bodov z laserového radaru Velodyne LiDAR.

Riešenie diplomovej práce vylepšuje už existujúce algoritmus, kde nahradzuje prahovanie príznakov za strojové učenie support vector machine. Veľká časť práce je zameraná na testovanie a porovnávanie starého a nového riešenia.

V kapitole o autonómnom robotovy Toad približuje jednu z možných platform a aplikácií analýzi okolia nie len pre autonómne vozidlá. Približuje aké ma Toad možnosti detekcie okolia. Táto kapitola tu je aj preto, lebo riešenie z ktorého táto práca vychádza, implementuje detektor pre túto platformu.

V kapitole o súčastných metódach priblíži pár rozličných postupov, ako danú problematiku riešiť. Analýzou polárnych histogramov? Použitím fuzzy clusterov? Detekovaním a trakovaním okrajov cesty? Rýchlim prahovaním základných vlastností z priestorových dát? Alebo použitie fúzie viacerých senzorov na riadenie autonomneho vozidla v púšti? Každý z prístupou má svoje silné a slabé stránky.

Teoretické znalosti sú priblížené v kapitole o support vector machine s postupným učením. Presnejšie základný prehľad SVM s postupným učením. Poslednou témou tejto kapitoly sú teoretické znalosti k vyhodnoteniu snímkov.

V návrhu riešenia a implementacií je spomenuté čo je predstavený použitý dataset. Čo všetko treba zmeniť v pôvodnom programe tak, aby bolo možné vyhodnocovať dáta z iného datasetu. A v neposlednom rade ako sa SVM učí.

V kapitole o vyhodnotení sú priblížené jednotlivé použité datasety, ich chyby a možnosti. Porovnanie priemernej úspešnosti na malom počte vzorkov je pri starom riešení 58% a pri novom 71%.

Kapitola 2

Autonómny robot Toad

Táto práca sa venuje rozpoznávaniu cesty pre robota, ktorý má osadený laserový radar Velodyne LiDAR. Na záznam dát a riadenie používa robotický framework ROS. Pre analýzu dát z radu je možné a vhodné použiť niektorú už z existújucih frameworkov napr. Point Cloud Library.

Velodyne LiDAR HDL-32E

Velodyne je americká spoločnosť, ktorá sa okrem iného venuje aj vývoju laserovým senzorom LiDAR. Jedným zo senzorov je práve HDL-32E.

HDL-32E využíva 32 laserov zarovnané medzi +10° a -30°, tak, aby poskytol kolme zorne pole. Senzor používa rotojúcu hlavu na frekvencii 10 Hz tak, aby v reálnom čase pokryla 360° horizontálneho zorné pole. Výstup z radaru je kontinuálne mračno bodov o 700 000 podov za sekundu. Dosah s presnosťou do 2 cm je 100 m [2].

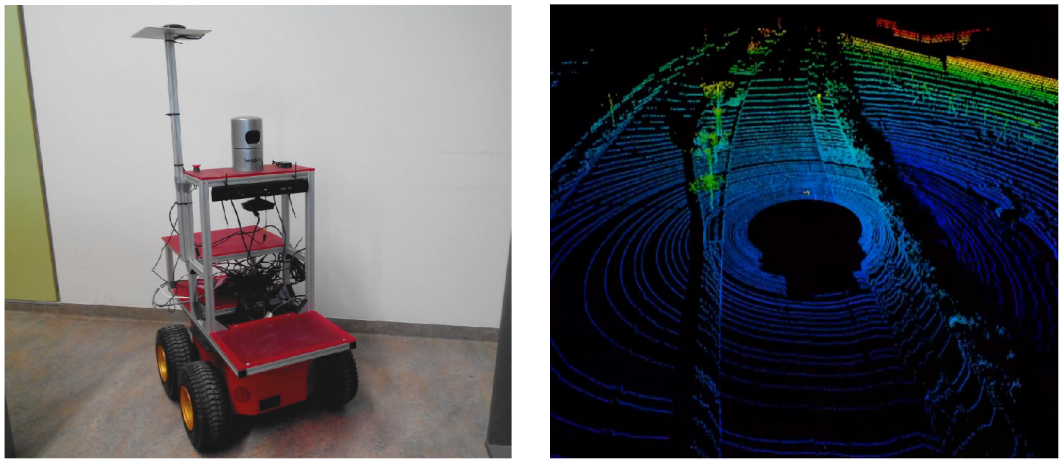
ROS – robotic operating system

Operačný systém pre robotov (ROS) je flexibilný framework na písanie robotického softwaru. Obsahuje zbierku nástrojov, knižníc a konvencii, ktorých cielom je zjednodušiť vývoj komplexných a robustných popisov správania robota naprieč širokou rozmanitosťou robotických platforiem [4].

ROS je open source pseudo-operačný systém, ktorý ponúka abstrakciu nad hardwarom, low-level prístup k riadeniu zariadení, implementáciu bežne používanej funkcionality a balíčkový systém. Komunikácia medzi procesmi je v ROS implementovaná rôznymi spôsobmi, ako synchronizovaný RPC cez služby (services), asynchrónny prúd dát cez tzv. témy (topics), alebo dáta uložené v tzv. parametrovom serveri [5].

Robot Toad

Toad je robotická platforma výskumnej skupiny Robo@FIT. Tento robot je zameraný na vývoj systémov na navigáciu, prehľadávanie, lokalizáciu a mapovanie v reálnom čase (simultaneous localization and mapping – SLAM). Toad (obrázok 2.1) disponuje senzorom Kinect ©. Tento senzor okrem RGB obrázku poskytuje aj hĺbkovu mapu. Ďalším senzorom je laserový radar Velodyne LiDAR, presnejšie model HDL-32. Na určenie polohy používa miniature GPS-Aided Inertial Navigation System. Všetky senzori sú v pohybe vďaka podvozku Pioneer 3-AT. Na jednotné riadenie sa používa pseudo-operačný systém ROS bežiaci vo verzii Hydro [12, 6].



Obr. 2.1: Ľavý obrázok je robot Toad. Pravý obrázok je príklad výstupného mračna bodov zo senzoru LiDAR [6].

Point Could Library

Framework Point Cloud Library (PCL) je projekt na prácu s mračnom bodov a na spracovanie 2D/3D obrázkov. V PCL je možné nájsť najvýkonnejšie algoritmi na filtrovanie, odhad príznakov, odhad povrchu, segmentácia, ... Tieto algoritmi sa dajú využiť na napr. na filtrovanie extremov zo zašumených dát, spájanie 3D mracien bodov dokopy, rozdelenie sceny na súvisiace časti, vytváranie povrchov, vizualizácia, ...

PCL je open source software, ktorý je možné použiť na rôznych platformách ako Linux, MacOS, Windows a Android / iOS [3].

Kapitola 3

Súčasné metódy pre detekciu cesty pre autonómne vozidlá

V tejto kapitole je priblížený prístup niektorých metód. Každá z nich používa iný prístup k preskúmaniu okolia cesty. Ako senzor je vždy zastupený laserový radar, či už v 2D alebo 3D verzií. Každá z následujúcich kapitol približuje inú metódu.

Road Terrain Detection: Avoiding Common Obstacle Detection Assumptions Using Sensor Fusion

Citované z [14].

Robustná metóda, ktorá spája viaceré vstupné senzori. Metóda zvláda detekciu prekážok vo veľkom množstve rozličných scenárov s použitím minimálneho počtu parametrov. Prístup je založený na analýze priestorových vzťahov získanych z jednej kamery a 3D LiDAR senzoru. Testovanie prebiehalo v rozličných podmienkach z datasetu ROAD-KITTI s uspokojujúcim výsledkami (viac ako 80%).

Metóda využíva spojenie riedkeho a neštrukturovaného 3D mračna bodov a obrázku (z kamery). Vyžaduje kalibračný krok k odhadu vnútorných a vonkajších parametrov. Z týchto je možné spočítať transformáciu medzi 3D mračnom bodov z reálneho sveta a zobraziť do 2D súradného systému obrázku. Hlavnou myšlienkou je využiť priestorové vzťahy v obrázku skombinované s 3D dátami k rozhodnutiu či bod odpovedá prekážke, alebo nie. Následne je vytvorený polárny histogram, ktorý je použitý na zostrojenie tzv. mapy dôvery. Táto mapa reprezentuje cestu v obrázku.

Celý systém pozostáva z postupného vykonávania piatich krokov:

- V prvom kroku sa tzv. spájajú dva vstupné senzori, čiže 3D mračno bodov sa premieta do obrazu.
- Druhý krok vytvára graf Delaunayovou trianguláciou[1], ktorý uchováva lokálne priestorové vzťahy medzi všetkými bodmi.
- V treťom kroku sa klasifikujú body či sú alebo nie sú prekážkou. Jediným ukazateľom je porovnanie výšky susedných bodov. (Obrázok 3.1)
- Štvrtý krok vytvára niekoľko polárnych histogramov, ktoré odhadujú voľné plochy.

[1] https://en.wikipedia.org/wiki/Delaunay_triangulation

- V poslednom – piatomdela kroku je vytvorená tzv. mapa dôvery z kombinácie všetkých voľných plôch z predošlého kroku.

Obr. 3.1: Výsledok detekcie prekážok. Zelené body reprezentujú tie prechody kde nie sú prekážky, modré body prechod medzi prekážkou a cestou, červené body prechod v prekážke.

Road Detection and Corner Extraction Using High Definition Lidar

Citované z [18].

Algoritmus na detekovanie a hľadanie okrajov vozovky v 3D mračne bodov. Analýza mračna bodov ako spôsob na rozpoznávania cesty, alebo iného objektu je časovo náročná úloha. Predstavovaný algoritmus je prispôsobený pre rýchle predspracovanie a jeho parametre nepotrebujú postupné učenie. Je to dosiahnuté použitím Fuzzy cluster metódy, založenej na rozdelovaní bodov podľa maximálnej entropie. Na rozdiel od niektorých metód filtre sú navrhnuté tak, aby pracovali s povrchom cesty a nie s jej okrajmi. Potom lineárne filtre viacnásobne vyvažované metódou najmenších štvorcov sú použité na rozlíšenie medzi lineárnym a nelineárnym segmentom rozložených bodov. Metóda na hľadanie rohov určí pozíciu budov na základe rozdielneho rozloženia bodov. Priestorové závislosti medzi rozdielnymi laserovými detektormi sú zohľadnené vo vysledných vlastnostiach.

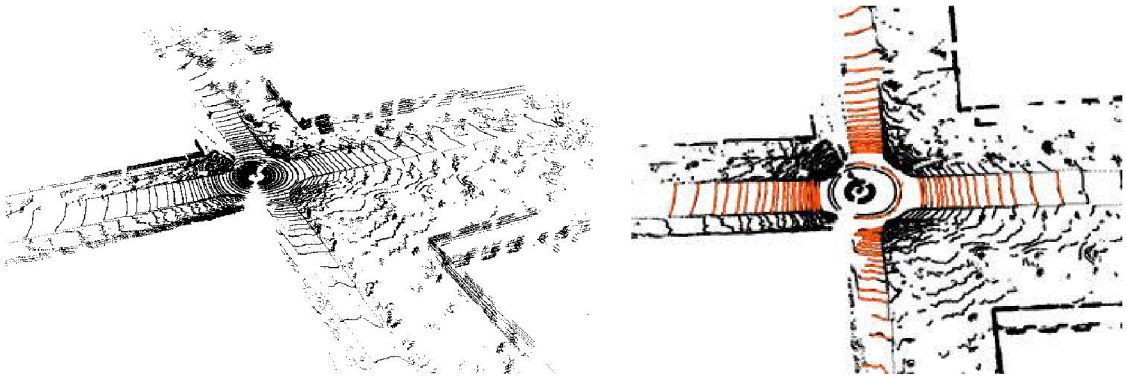
Keď robot naviguje v metskom prostredí automaticky, musí nájsť cestu a jej ohraničenie. Ako senzor bol zvolený Velodyne LiDAR HDL-64E. Získavanie príznakov prebieha pre každú skenovanú líniu zvlášť, následne sa vyváži napriek 64 laserovým skenom v jednej linii. Týmto spôsobom algoritmus funguje rýchlejšie než keby sa klasifikovalo celé mračno bodov.

Road boundary detection and tracking for structured and unstructured roads using a 2D LiDAR sensor

Citované z [10].

Algoritmus na detekovanie a následné sledovanie okrajov cesty pre spevnené aj nespevnené cesty. Algoritmus získa príznaky cesty ako úsečky v polárnom súradnom systéme. Tieto príznaky sa sledujú v závislosti na súradnom systéme vozidla s použitím tzv. "nearest neighbor"filtra. Tento algoritmus presne detekuje okraje cesty bez ohladu na typ cesty.

Systém sa skladá z týchto troch častí. Laserový senzor LiDAR vytvára dáta v polárnych súradniciach. Z týchto dát sú získané úsečky. Tieto úsečky sú korigované voči výške a natočeniu senzora.



Obr. 3.2: Detekcia cesty. Ľavý obrázok reprezentuje výstup z LiDAR HDL-64E. Pravý obrázok výsledok: oranžové línie predstavujú cestu [18].

Každá z priamok je ohraničená ľavý a pravý okraj. Ľavý a pravý okraj je meraným výstupom na sledovanie ľavého a pravého okraju cesty. Každý z okrajov používa na určenie pozície algoritmus Nearest Neighbor Filter.

2D lidar dáva dáta v polárnych súradniciach, tieto dáta sa ďalej nekonvertujú, ale používajú nezmenené v metóde na hľadanie rovných úsekov – priamok. Táto metóda prihliada aj na výšky, uhol náklonu a uhol stúpania terénu. Metóda na vyhľadávanie priamok je založená na Iterative-End-Point-Fit (IEPF) algoritme[2]. Body, ktoré nie sú na pomyselnej úsečke sa využívaju v algoritme na určenie hľadaných sekvencí. Následne sú získané úsečky v každej postupnosti dát z LiDARu.

Hľadanie ohraničení sa používa adaptatívný algoritmus, jeho úlohou je nájsť nelinearitu vo vzdialenostných dátach ρ. Ak je rozdiel ρ_i a ρ_{i-1} väčší než adaptatívny prah. Tento prah je odvodený z ρ_{i-1} s korekciou na presnoť senzora, šum a najhorší možný uhol odvodený z pre ρ_i.

Výsledky pre nespevnené cesty ukázali, že napriek nerovnému povrchu cesty a rýchlosti vozidla, ktorá bola počas testu od 30 km/h do 50 km/h, po ktorom vozidlo išlo, tak účinnosť metódy bola nad $92,0\%$ pri detekovaní cesty s menej nez $0,8\%$ nesprávnou pozitívnou detekciou. Výsledok tohto experimentu je možné vidieť na obrázku 3.3

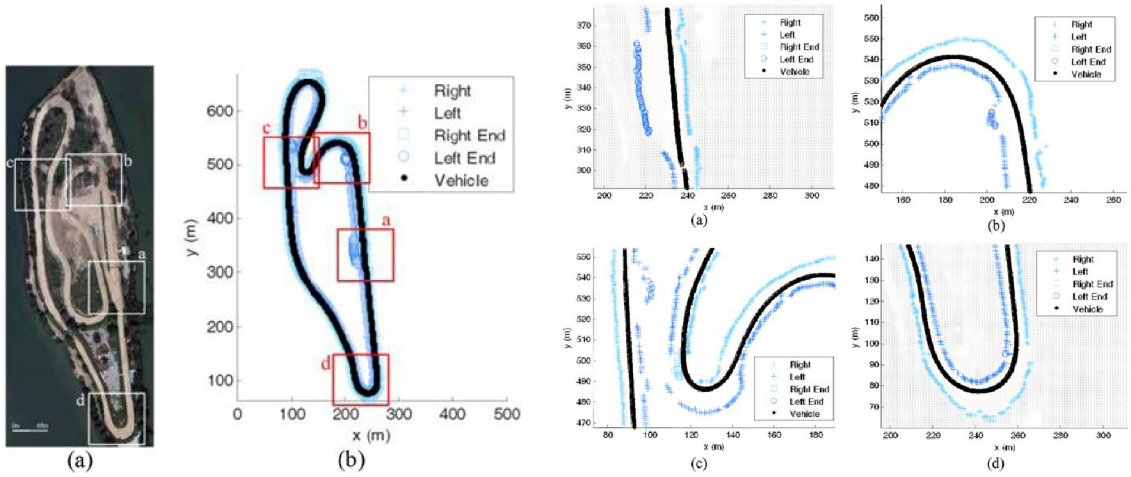
Spevnené cesty sa rozdelovali na dva typi, tie ktoré boli ohraničené obrubníkmi a tie ktoré nemali výrazný rozdiel vo výške ohranicenia cesty. Cesty s obrubníkmi v prvom testovacom scenáry dosiahli aspoň $81,4\%$ úspešnosť detekcie s menej než $2,7\%$ nesprávnou pozitívnou detekciou. Druhý scenár dosiahol úspešnosť aspoň $85,8\%$ s menej než $1,1\%$ nesprávnou pozitívnou detekciou. Spevnené cesty s obrubníkmi dosiahli úspešnosť aspoň $95,7\%$ s menej než $4,5\%$ nesprávnou pozitívnou detekciou.

RoadCompass: following rural roads with vision + ladar using vanishing point tracking

Citované z [8].

Prístup k riešeniu autonómneho vozidla na rustikálnych cestách a na púšti, ktoré sú riadené na základe analýzi dát z LiDARu a kamery, boli rozsiahle testovania na uzavretých systémoch. Analýza z kamery je založená na Gáborovych vlnkových filtroch. Tieto filtre

[2]Borges and Aldon, 2004; Duda and Hart, 1973

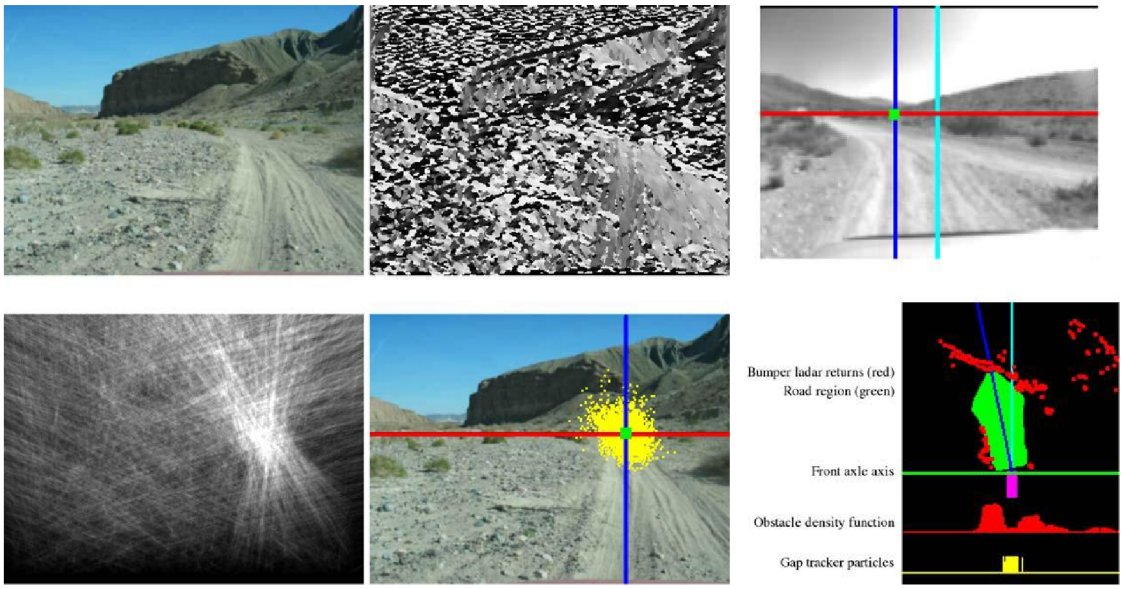


Obr. 3.3: Výsledok algoritmu pre nespevnené cesty. Ľavý obrázok ukazuje trénovací okruh. Pravý obrázok sporné detaily, kde chyba bola najväčšia [10].

vyhodnocujú textúru tak, aby našli kolaje, alebo stopy po vozidle, ktoré už tadiaľto prechádzalo. Z výstupu tohto filtru sa dá odvodiť obraz úbežného bodu (vanishing point) a to z výsledku Houghofho hlasovania sa získa odhad smeru pre ovládanie riadenia. Analýza dát z LiDARu slúži k detekcii prekážok, ktoré sú v bezprostrednej blízkosti ku smeru jazdy vozidla a následnej korekcí tak, aby vozidlo sa pohybovalo po ideálnej stope. Tento proces sa snaží priblížiť obrázok 3.4.

Metóda obsahuje tri významné kroky k odhadu tvaru cesty.

- Analýza textúry je vykonaná pomocou výpočtu dominantnej orientácie textúry v aktuálnej snímke.
- Lineárny odhad smeru cesty je zvolený na základe hlasovania všetkých orientovaných textúr pre najlepší ubežník.
- Určenie bočnej odchýlky dráhy od stredu cesty a šírka cesty sú vypočítané z LiDAR dát.



Obr. 3.4: Postup algoritmu pri určovaní cesty. Prvé štyri obrázky v ľavo znázorňujú postup na nájdenie najlepšieho úbežníka a to od obrázku z kamery, cez zobrazenie textúry dominantných hrán. Tretí obrázok ukazuje výstup z Gáborových vlkových filtrov. Posledný zo štyroch obrázkov v ľavo ukazuje výsledok hlasovania orientovanách textúr. Ostatné dva obrázky v pravo ukazujú spojenie výstupu z analýzi obrázka a laserového senzoru [8].

Kapitola 4

Súčastné riešenie pre robota Toad

Celá kapitola napísaná podľa zdrojového kódu z [1].

Ďalším existúcim riešením je skupina aplikácií pre spracovanie dát z laserového radaru Velodyne Lidar. Toto riešenie bude brané ako základ pre ďalší vývoj aplikácie pre túto diplomovú prácu.

Projek založený vedeckou skupinou Robo@FIT implementuje tri aplikácie Laser Scan, Cloud Assembler a Ground Map. Každá z nich je nezávisle spustiteľná a použiteľná. Celý balík aplikácii sa volá but_velodyne_proc.

Cloud Assembler spája viaceré snímky radaru do jedného obrázku. Týmto dosahuje hustejší model. Aplikácia Laser Scan simuluje konvečný 2D skener.

Poslednou zo spomínaných je bližšie priblížná v nasledujúcich riadkoch.

4.1 Ground map

Ground map je aplikácia, ktorá na vstupe očakáva Velodyne 3D sken a výstupom je mapa okolia robota s bezpečnou cestou. Algoritmus na zistenie bezpečnej cesty používa dva prístupy. Body z velodyne v okolí robota sú rozdelené do buniek. Na bunku sa dá pozerať ako na najmenší prvok z ktorého sa odvodzuje cesta. Z týchto buniek sa spočítajú príznaky. Na základe vyhodnotenia príznakov je vyhodnotené či bunka je, alebo nie je cesta. Blokovú schému približuje obrázok 4.1.

Ako vstup do algoritmu je štruktúra *velodyne_pointcloud::PointXYZIR*. Každý bod je reprezentovaný troma súradnicami, intenzitou a určením čísla prstenca. Číslo prstenaca sa používa pri vyhodnocavaní príznaku hranatosti.

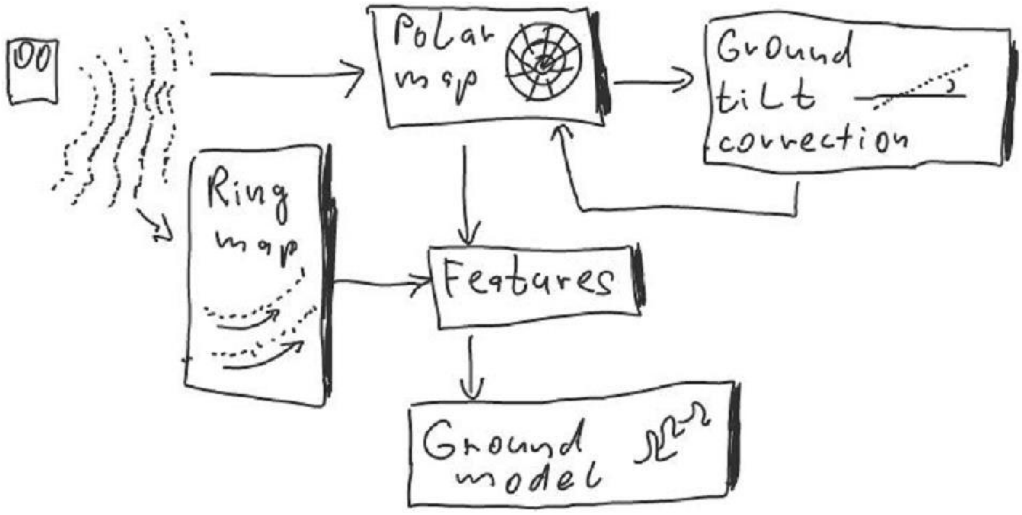
Ako prvý krok treba odhadnúť zem. Tento odhad základovej dosky je vykonaný za pomoci point cloud library a RANSAC algoritmu. Od zeme sa odvodzuje výška bodov (súradnica z).

Algoritmus prechádza bunky v dvoch smeroch. Po prstencoch a polárne. Po prstencoch zistuje príznak hranatosť (obrázok 4.2), polárne zase príznak drsnosť (obrázok 4.3), posledným použitým príznakom je odrazivosť.

Model cesty je popísaný troma gausovými funkciami. Každá z funkcí je odvodená z buniek, ktoré by mohli byť cestou. Tento odhad je založený na minimálnom počte vzorkov v bunke a parametre najvacšieho rozdielu výšky medzi bodmi v bunke.

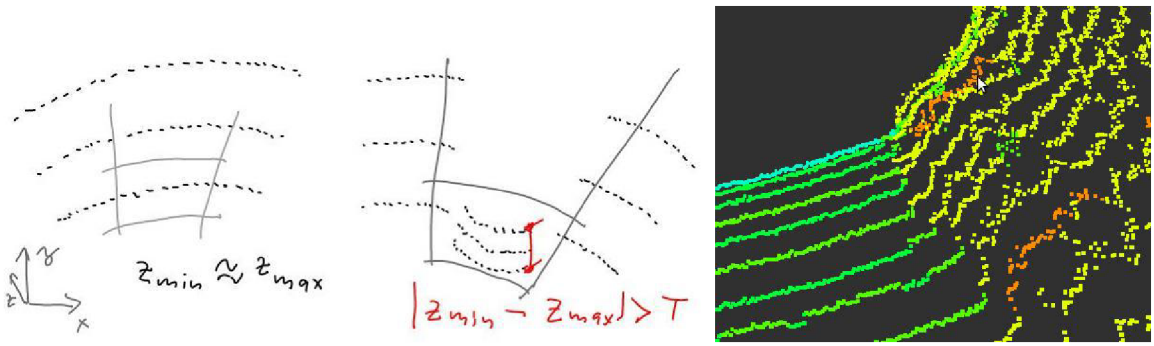
Príznaky použité pre výpočet modelu cesty:

- Drsnosť (roughtness)



Obr. 4.1: Bloková schéma metódy [13].

- Intenzita povrchovej odrazivosti (reflectivity)
- Hranatosť (edginess)



Obr. 4.2: Príznaky získané z analýzi polárnej mapy. Lavé dva obrázky ukazujú príznak drsnosť. Na pravom obrázku je vidieť povrchová odrazivosť (zmena farby bodu), tieto dáta sú k dispozícii priamo od Lidaru [13].

Príznak drsnosť je pre bunku charakterizovaný ako rozptyl výšky od zeme:

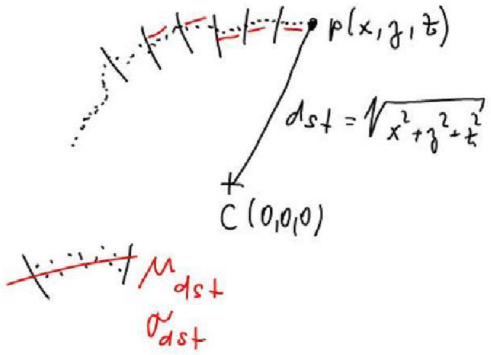
$$G_{var} = \frac{1}{N}\left(\sum_{i=1}^{N}(p_z^i)^2 - \frac{1}{N}\left(\sum_{i=1}^{N} p_z^i\right)^2\right) \tag{4.1}$$

kde N je počet bodov v bunke, p_z je súradnica z bodu p.

Príznak odrazivosť je pre bunku charakterizovaný ako priemerná výška od zeme:

$$R_{sum} = \frac{1}{N}\sum_{i=1}^{N} p_{\text{intesity}}^i \tag{4.2}$$

kde N je počet bodov v bunke, p_z je súradnica z bodu p.



Obr. 4.3: Príznaky získané z analýzi hrantosti [13].

Pre príznak hranatosť sa body podľa informácie z mračna bodov rozdelia do prstencov. Pre rozdelenie sa použije informácia o indexe prstenca. Prstence sa ďalej delí do buniek, toto delenie je už na základe zvolených uhlú.

Najprv je treba vypočítať pozíciu bunky voči radaru (vzťah 4.3).

$$\text{bin}_{\text{pos}} = min\left(\sqrt{(p_x^i)^2 + (p_y^i)^2 + (p_z^i)^2}\right) \quad \Big| \quad i \in 1 \ldots N \tag{4.3}$$

kde N je počet bodov v bunke, p_x je súradnica x bodu p.

Hranotosť bunky je potom porovnanie dvoch susedných bodov:

$$E = \left|\text{bin}_{\text{pos}}^{\text{current}} - \text{bin}_{\text{pos}}^{\text{next}}\right| \tag{4.4}$$

kde currect je aktuálna bunka, next je susedná bunka na rovnakom prstenci.

Gausové funkcie popisujúce cestu z vyššie spomenutých príznakov:

$$\mu_R = \frac{1}{K}\sum_{j=1}^{K} R_{var}^j \tag{4.5}$$

$$\sigma_R = \sqrt{\frac{1}{K}\left(\sum_{j=1}^{K}(R_{var}^j)^2 - \frac{1}{K}\left(\sum_{j=1}^{K} R_{var}^j\right)^2\right)} \tag{4.6}$$

kde K je počet buniek z ktorých sa počítali príznaky.

Vzorce pre gausové funkcie ostatných príznakov sú takmer identické.

Po určení modelov sa vyhodnotí pravdepodobnosť, či bunka je cesta. Výsledná pravdepodobnosť je len súčinom pravdepodobností modelov príznakov.

Prahovanie modelov je založené na dvoch faktoroch. Výška bodov v bunke a príslušnosť/pravdepodobnoť či bunka patrí do modelu. Výška je vzialenosť od zeme, ktorá sa porovnáva s prahom a vstupným parametrom vyjadrujúcu maximálnu nepravidelnosť cesty. Príslušnosť je súčin výsledok Gausových funkcii modelu, ktorá musí byť vačšia než zvolený prah.

Výsledná mapa úspešne detekuje hrany prekážok, ale výsledkom pre vačší počet buniek býva vyhodnotený ako neznámy.

Slabinou tohto prístupu sú vopred určené prahy, ktoré nedovolujú prispôsobenie sa zmene okolia. Čiže napr. ak by robot mal prejť po prašnej ceste ale prahy by boli nastavené na rovný asfaltový povrch.

Kapitola 5

Support Vector Machine s postupným učením

Support Vector Machine

Support Vector Machine (SVM) je jeden z modelov strojového učenia, ktorý sa zväčša trénuje tzv. s dohľadom (supervised learning) [7]. Najčastejšie využitie nájde v klasifikácii a regresnej analýze. Vstupom je trénovacia sada, kde pre každú vzorku je vopred známa príslušnosť do jednej z dvoch kategórii. SVM potom vytvorí model, ktorý je schopný priradovať nové prvky do jednej z dvoch kategórii. Preto sa táto metóda radí medzi nepravdepodobnostné binárne lineárne klasifikátory. Ako rozšírenie, je tiež možné použiť nelineárnu klasifikáciu pomocou využitia jadra (kernel).

Základná verzia metódy pracuje s vopred známou množinou trénovacích dát. Po spracovaní celej trénovacej množiny je možné použiť model ku klasifikácii. Tento prístup sa volá offline. Existuje ale aj postupné učenie.

SVM s postupným učením

Postupné učenie je výhodné ak sa musíme vysporiadať s veľmi veľkými alebo nestálymi (non-stationary) dátami. V prípade nestálych dát, by učenie na celej konečnej množine obecne zlyhalo, ak by sa napr. rozdelenie sa menilo v čase. Na veľa zaujímavých problémov sa v strojovom učení môže hľadieť ako na vhodné pre postupné učenie [15]. Praktická výhoda tohto prístupu je, že umožňuje zahrnúť ďalšie trénovacie dáta vtedy, keď sú dostupné a to bez toho aby bolo potrebné opätovne celé pretrénovať. Je dané, že trénovanie je zvyčajne výpočetne najnáročnejšie a nikoho neprekvapí, že prístupnosť algoritmov, ktoré sa dokážu postupne učit sú vhodné pre prácu s veľkým množstom dát. Ďalším dobrým príkladom je odhad v reálnom čase pre nepretržitý prúd dát, ako napr. dolovanie dát z web-u alebo rozhranie medzi mozgom a počítačom.

Pegasonov algoritmus

Citované z [17].

Jednoduchý a efektívny algoritmus trénovania SVM, ktorý pracuje na princípe stochastického sub-gradient descent pre riešenie optimalizacných problémov nachádzajúcich sa v SVM. Je dokázané, že počet iterácii potrebných k získaniu výsledku s presnosťou ϵ je

$\widetilde{O}(1/\epsilon)$, kde každá iterácia pracuje s jednoduchým trenovacím príkladom. Na rozdiel s predchádzajúcou analýzou stochastických gradient descent metód pre SVM vyžaduje $\omega(1/\epsilon^2)$ iterácii. V predchádzajúcom výskume dokázali, že počet iterácii rastie lineárne s $1/\lambda$, kde λ je regularizačný parameter SVM. Pre lineárne jadro, je celkový čas behu v tomto algoritme určený $\widetilde{O}(d/(\lambda\epsilon))$, kde d je počet nenulových príznakov v každom z príkladov. Pretože čas behu algoritmu priamo nezávisí na veľkosti trénovacej sady, tak je vhodný na použitie pre veľké datasety. Tento algoritmu umožnuje použitie nelineárnych jadier, ale v tomto prípade čas behu algoritmu lineárne závisí od veľkosti trénovacej sady.

Po analýze efektívnosti algoritmu sa zistilo, že pre lineárne jadrá Pegasos dosahuje výrazne lepšie výsledky ako konkurenčné algoritmy. Pri použití komplikovanejších jadier, algoritmus je stále porovnateľný so zložitejšími prístupmi.

Metóda na vyhodnotenie úspešnosti klasifikácie

Na vyhodnotenie úspešnosti klasifikácie algoritmu je použité binárne štatistické ukazatele presnosť a citlivosť (*Precision & Recall*).

Tieto ukazatele zavádzajú pojmy ako:

- pravdivo pozitívne (*true positives*)
- pravdivo negatívne (*true negatives*)
- nepravdivo pozitívne (*false positives*)
- nepravdivo negatívne (*false negatives*)

Pravdivo pozitívne a pravdivo negatívne sú snímky, ktoré boli správne určené, v tomto prípade ak cesta bola vyhodnotená ako cesta a prekážka ako prekážka. Nepravdivo pozitívne sú zase tie body, ktoré sú prekážkou, ale boli vyhodnotené ako cesta. Nepravdivo negatívne sú body, ktoré boli vyhodnotené ako prekážka, ale pritom sú cestou.

Presnosť (5.1) vyjadruje vzťah medzi bodmi, ktoré boli správne vyhodnotené ako cesta a všetkými vybranými bodmi. Citlivosť (5.2) určuje koľko cesty bolo správne určených.

Daľší ukazateľ určuje mieru s akou presnosťou boli určené prekážky – *true negative rate* (5.3). Je to rovnaký ukazateľ ako citlivosť, len pre negatívne vzorky. Ukazateľ *accuracy* (5.4) ukazuje, koľko z celého počtu snímkov bolo správne klasifikované.

F-measure (5.5) je harmonickým priemerom ukazateľov presnosť a citlivosť.

$$\text{Precision} = \frac{tp}{tp + fp} \tag{5.1}$$

$$\text{Recall} = \frac{tp}{tp + fn} \tag{5.2}$$

$$\text{True negative rate} = \frac{tn}{tn + fp} \tag{5.3}$$

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn} \tag{5.4}$$

$$\text{F-measure} = 2\,\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.5}$$

Kde tp sú snímky, ktoré boli správne vyhodnotené ako cesta. tn snímky, ktoré boli správne vyhodnotené ako prekážka. fp snímky, ktoré boli nesprávne vyhodnotené ako cesta. fn snímky, ktoré boli nesprávne vyhodnotené ako prekážka.

Kapitola 6

Návrh detekcie cesty s využitím Velodyne LiDAR a SVM

Úlohou tejto práce je vylepšiť už existujúci postup. Ako základ mojej práce chcem použiť už existujúce riešenie opísané v kapitole 4 a následne ho rozšíriť. Zdrojový kód už existujúceho riešenia je uložený na GitHub[1].

Rozšírenie, nebodaj zlepšenie pôvodnej implementácie bude pozostávať z nahradenia prahovania príznakov za komplexnejšie riešenie a to strojové učenie. Ako metódu strojového učenia som si zvolil Support Vector Machines.

Aplikácia je samostatne spustiteľným ROS uzlom. Uzol komunikuje cez jednu vstupnú a jednu výstupnú tému. Aplikácia čaká na nadviazanie oboch prúdov komunikácie a až potom začne spracovávať vstup a poskytovať výstup.

Vstup a výstup aplikácie

Vstupom do aplikácie je mračno bodov typu *sensor_msgs::PointCloud2*. Vstup má tému s názvom *clound_pcd*.

Výstupom je mapa dátového typu *nav_msgs/OccupancyGrid*. Výstup má tému s názvom *map2d_out*.

Spracovanie vstupného mračna bodov prebieha bez *tf* synchronizácie po jednotlivých snímkoch cez callback funkciu *GroundMap::process*. Funkcia spracováva vstup nezávisle (až na stav SVM) od predchodzích snímkov.

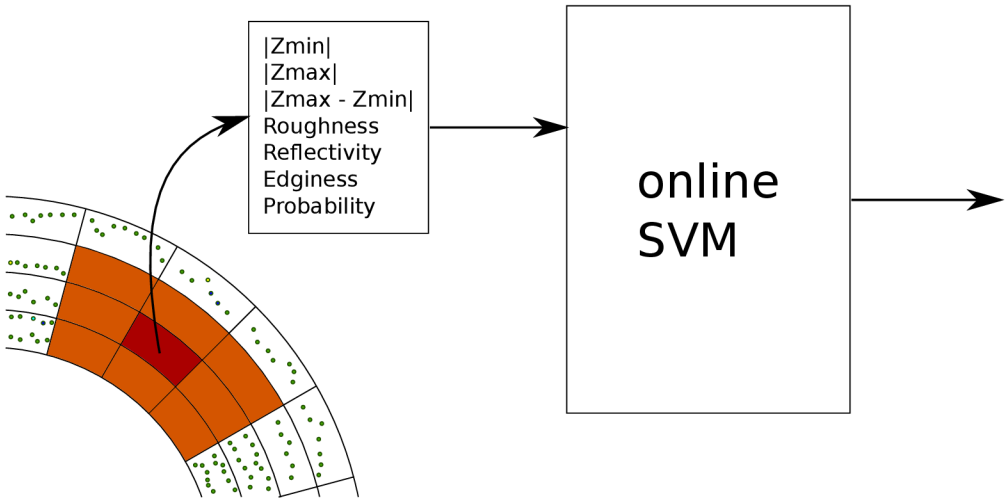
Pôvodný algoritmus musel byť upravený tak, aby pracoval so vstupným štruktúrou XYZIA namiesto XYZIR. Pre každý bod štruktúra XYZIA obsahuje súradnice x, y, z, následuje intenzita a anotácia. Narozdiel od štruktúry XYZIR, ktorá neobsahuje anotáciu ale číslo prstenca. Keďže body z KITTI datasetov neobsahovali informáciu o čísle prstenca, tak túto informáciu bolo treba dopočítať zo samostatných súradníc bodov.

Index prstenca nie je zložité zistiť, keďže z dokumentácie k LiDAR-u je možné vyčítať počet indexov, maximálny a minimálny uhol pod ktorým sú lúče vysielané. Z toho sa dajú jednoducho určiť intervaly uhlov a ich odpovedajúci index. Výpočet uhlu τ vyslaného luču je pre každý bod $\tau = \tan \frac{z}{\sqrt{x^2+y^2}}$.

[1] https://github.com/robofit/but_velodyne/tree/hydro-devel/but_velodyne_proc

SVM

Po vyhodnotení všetkých príznakov sa spracované bunky sa prechádzajú postupne a vhodné učia SVM. Výber vhodných buniek, ktoré je potrebné SVM naučiť je vyberať len tie extréme. Čiže len tie, ktoré by sa dali jednoducho opísať ako najrovnejšie, alebo najčlenitejšie, ciže práve tie, ktoré najvačšou pravdepodobnosťou boli, alebo neboli cestou.



Obr. 6.1: Schéma zapojenia online SVM.

Extrémne bunky sa dajú popísať príznakmi cesty, ako drsnosť (vzťah 4.1), hranatosť (vzťah 4.5) a odrazivosť (vzťah 4.2). Tie bunky, ktoré majú výslednú pravdepodobnosť, čiže súčin príznakov vačší než zvolená minimálna pravdpodobnosť pre cestu sú trénované ako cesta. Naopak tie bunky, ktoré majú pravdepodnosť menšiu než zvolenú maximálnu pravdepodobnosť pre prekážku sú trénované ako prekážka.

Ako je na obrázku 6.1 vidieť, vstupom do SVM nie je nikdy len práve analyzovaný bod, ale aj jeho okolie. Okolie je vypočítané s polárnych súradníc a sú vyberané susedné bunky. Týmto dosiahneme, že SVM dostane väčšie povedomie o okolí a môže sa naučiť aj na prvý pohľad tažko určiteľné vzťahy.

Príznaky, ktoré sa používajú pre trénovanie online SVM sú:

- $|z_{min}|$
- $|z_{max}|$
- $|z_{max} - z_{min}|$
- roughness
- reflectivity
- edginess
- probability(roughness $\cdot$ reflectivity $\cdot$ edginess)

Výstupom SVM je určenie či spracovávaná bunka je, alebo nie je cesta.
Čiže 7 príznakov pre každú z 9 buniek.

Normalizácia snímkov

Pred učením je potrebné príznaky normalizovať. Normalizácia snímku je vyjadrená vzťahom 6.3:

$$\bar{p} = \frac{1}{N}\sum_{i=1}^{N} p^i \tag{6.1}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p^i - \bar{p})^2} \tag{6.2}$$

$$p_n^i = \frac{p^i - \bar{p}}{\sigma} \qquad i \in 1 \ldots N \tag{6.3}$$

Kde N je počet vzorkov. Vzťah 6.1 je priemer $\bar{p}$ príznakov p. Vzorec 6.2 je smerodajná odchýlka σ.

Dôležitým faktorom pre učenie je vyvažovanie pozitívnych a negatívnych vzorkov.

KITTI dataset

KITTI je komplexny dataset, ktorý je používaný aj k benchmarkom. Hlavným zameraním KITTI data setov je poskytnutie voľných dát na vývoj v odvetviach počítačového videnia ako stereo kamera, optický tok, vizuálnu odometriu, detekcia 3D objektov a 3D sledovanie objektov. Tieto datasety obsahujú pomerne širokú škálu záznamov z rôznych prostredí a to od mestských ulíc (obrázok 6.2) a križovatie, vidieckych ciest až k záznamom z dialnice. Dataset je zaznamenaný z bežného auta pomocou štyroch kamier (dve nahrávajúce záznam v odtienoch sivej, dve farebné) so záznamom vo vysokom rozlíšení, jeden laserovy radar Velodyne Lidar vo verzií HDL-64E a inerciálny systém (GPS/IMU). Radar je osadený vo výške $1,73\,\mathrm{m}$.

Obr. 6.2: Príklad dát z KITTI datasetu.

KITTI ponúka [9] všetky záznamy s týmito informáciami:

- dáta ako v RAW formáte (nesynchronizované a neopravené), tak aj ich synchronizované a opravené verzie.
- Kalibrácia medzi senzormi (kamera, kamera - GPS, kamera - Velodyne).
- Označené 3D objekty (autá, nakladné autá, električky, chodci, cyklisti).

Kapitola 7

Implementácia riešenia

Implemetácia výchadza a rozširuje už existuje riešenie. Implementacia v ROS vo verzií Groovy.

Vývoj aplikácie bol vykonaný na dátach nahraných Toadom 2. Potom bolo treba aplikáciu upraviť tak, aby bolo možné načítať dáta v inom formáte. Nový formát vyžaduje vypočítať index prstenca. Index prstenca sa počíta pre každý bod zvlášť.

Ako druhá zmena bolo treba zmeniť scénu, čiže veľkosť analyzovanej časti. Parametre ktoré som zvolil:

- angular_res = 3
- radial_res = 0.3
- min_range = 3.7
- max_range = 20

7.1 Dlib

Citované z [11].

Ako implementáciu Pegasonovho algorimu som zvolil dlib. Dlib je open-source knižnica, zameraná ako na inžinierov tak aj na vedcov, ktorý chcú bohaté prostredie. Knižnica umožnuje jednoduchý vývoj algortimov postavených na strojovom učení v programovacom jazyku C++. K splneniu tohto cielu dlib obsahuje rozšíriteľnú lineárnu algebru s podporou BLAS[1]. Implementuje algoritmi na prácu s Bayesianovimi sieťami a metódmi pracujúcimi s jadrami (kernel-based methods) na rozpoznávanie, regresiu, zhlukovanie a detekciu anomálii. Knižnica má kompletnú dokumentáciu a nastroje pre debugovanie.

Z knižnice Dlib som využil práve implementáciu SVM pegasos.

SVM s postupným učením

Odhad parametrov pre vyhľadávanie extrémnych buniek bol vykonaný pomocou experimentálne odhadnúť:

- max_road_irregularity = 0.04

[1] Basic Linear Algebra Subprograms

- max_height_diff = 0.025
- ground_prob = 0.95
- obstacle_prob = 0.2

Na online SVM som použil implementáciu *dlib::svm_pegasos* z knižniče *dlib/svm.h*.

Experimentálne som zistil, že pri zvolených príznakoch mi najlepšie výsledky dáva SVM kernel: *dlib::radial_basis_kernel*. Pre tento kernel som zvolil parametre $\gamma = 0.005$ a $\lambda = 0.00001$. Pre aspoň pár snímkov za sekundu, som ale musel zvoliť nižší počet SVM na 20.

Normalizovanie vzorkov pomocou *dlib::vector_normalizer*. V prvom prechode bunkami hľadám ich okolie a ukladám príznaky do vektoru. Zozbierané príznaky uložené vo vektori normalizujem.

Pri učení vyvažujem vzorky podľa počtu naučených buniek cesty a buniek prekážky. Presnejšie ak je počet buniek naučených vačší ako naučených buniek prekážky $bin_{road} + \frac{bin_{road}}{8} \geq bin_{obstacle}$. Tak je možné učiť.

Kapitola 8

Meranie

Vyhodnotenie úspešnosti rozšírenej implementácie je vykonanné na anotovanom datasete. Ciže na datasete, ktorý obsahuje snímky mračna bodov a informáciu o tom, či daný bod je zem alebo prekážka. Pre porovnanie potencionálneho zlepšenia je zmeraný aj pôvodný algoritmus.

Dataset, ktorý ponúka ako záznam z LiDAR-u, tak aj anotácie k mračnu bodov sa volá KITTI.

8.1 Anotované datasety a ich chyby

Na webovej stránke KITTI v sekcií sémantika je možné nájsť aj anotáciu pre Velodyne dáta[1][16]. Celkovo je anotovaných 8 datasetov, kde sú objekty rozdelené do 11 kategórií: 1 – budovy, 2 – obloha, 3 – cesta, 4 – vegetácia, 5 – chodník, 6 – auto, 7 – chodec, 8 – cyklista, 9 – značky, 10 – plot a 0 – neanotované.

Nie všetky kategórie sú použité pre tie body na ktoré nie je vidieť z pohladu kamier. Pre testovanie úspešnosti detekcie cesty som použil body kategórie 3. Zvyšné kategórie boli označené ako prekážka.

Všetky anotované datasety sú z urbanistického prostredia. Na záznamoch je vidieť prechod križovatkou, štvorprúdová cesta rozdelená plotom, prechod po ceste popri zaparkovných aut. Všetky tieto snímky ponúkajú zaujímavé scény pre vyhodnotenie.

K vyhodnoteniu som použil snímky zo štyroch datasetov. Dva sú z mestského prostredia, a po jednom z cesty a obytného prostredia.

Metské prostredie je zastúpené dvoma datasetmi:

- 2011_09_26_drive_0005 (obrázok 8.1)
- 2011_09_26_drive_0014 (obrázok 8.2)

Prvý spomínaný záznam má 16 sekúnd a obsahuje 16 anotovaných snímkov z celkového počtu 154 snímkov. Na zázname je vidieť deväť áut, tri dodávky, dvoch chodcov a jedného cyklistu. Na zázname je vidieť prechod metskými uličkami a troma križovatkami.

Záznam 0014 má 32 sekúnd. Anotovných je 31 snímkov z celkového počtu 318 snímkov. Na zázname je vidieť dvadsaťšesť áut, štyri dodávky, jedno nákladné vozidlo, päť chodcov, štyroch cyklistov a jednu električku. Záznam ukazuje širšie a rovné cesty.

[1] http://www.cvlibs.net/datasets/kitti/eval_semantics.php

Obr. 8.1: Obrázok z datasetu 0005

Obr. 8.2: Obrázok z datasetu 0014

Obytné prostredie (obrázok 8.3) je zachytené na anatovanom datasete 2011_09_26_drive_0036. Dĺžka tohto záznamu je 80 sekúnd. Z celkového počtu 803 snímkov je anotovaných 41. Na zázname je vidieť osemdesiat áut, sedem dodávok, jedno nákladné vozidlo a jedného chodcu. Na zázname je vidieť prechod z obytného prostriedia s veľkým množstvom zaparkovaných áut až k ceste lemujúcu železničnú trať.

Anotovaný cestný záznam 2011_09_26_drive_0015 má 30 sekúnd, 297 radarových snímkov a 30 anotovaných. Na zázname je vidieť tridsaťtri áut, jednu dodávku, jedno nákladné vozidlo a jedného cyklistu. Záznam obsahuje rovnú cestu s jedným nespevneným okrajom vozovky (obrázok 8.4).

Ďalšie anotované datasety sú z prostredia nevhodného pre rozpoznávanie len pomocou laserového radaru. Sú vyhotovené na okraji peších zôn, kde sú síce chodníky, ale nie je ich možné rozlíšiť len na základe výšky.

Pri bližšom preskúmaní anotacií som zistil, že nie všetky snímky sú úplné dotiahnuté do dokonalosti. Hlavnými nedostatkami sú:

- Pásy zle anotovaných bodov v niektorých po sebe idúcich snímkach (obrázok 8.5 vľavo).
- Chybná anotácia menších prekážok napr. ostrovček okolo značky (obrázok 8.5 vpravo).
- Autá do určitej výšky boli označené ako cesta (obrázok 8.6).
- Chodník, pás zelene pri zvodidlách je označený ako cesta (obrázok 8.7).

Obr. 8.3: Obrázok z datasetu 0036

Obr. 8.4: Obrázok z datasetu 0015

- Na okrajoch zatienených miest autom, alebo senzormi sú dáta veľmi nepresné.

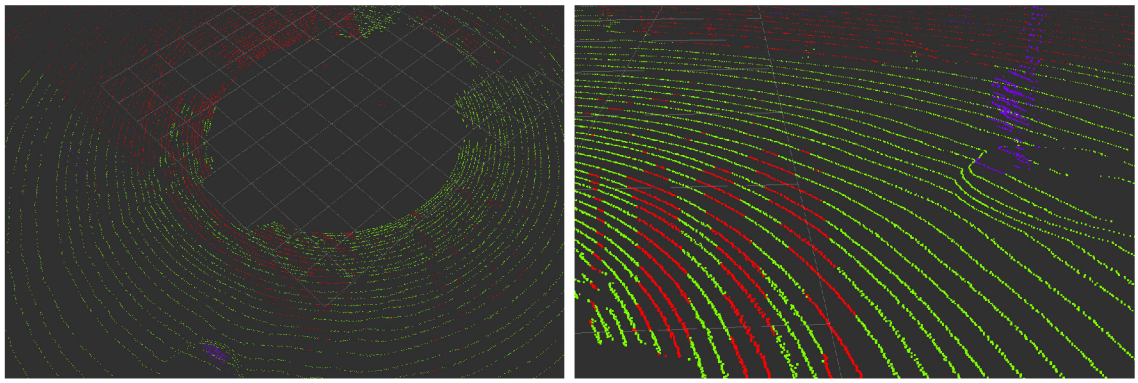
Tieto chyby zťažuhú učenie s učiteľom, kedže do SVM vstupujú aj zle anotované vzorky. Čiže aj vzorky ktoré obsahujú obrubník, ale sú anotované ako cesta, alebo naklonený členitý povrch.

Vyhodnotenie nie je o nič lepšie, keďže vzorky, ktoré niesú cestou sú označené ako cesta. Príkladom môže byť obrubník oddelujúci chodník od cesty. Ak je tento obrubník označený ako cesta, tak algoritmus vyhodnotí obrubník ako prekážku, čo je správne, ale anotácia hovorí, že toto je stále cesta. Druhým častým problémom je, že chodník je niekedy rovnejší ako cesta. Tento problém by sa dal vyriešiť prísnejším prahom na výšku od zeme. Keďže ale pracujeme s reálnymi dátami, tak problém by nastal pri stúpaní väčšom ako daný prah.

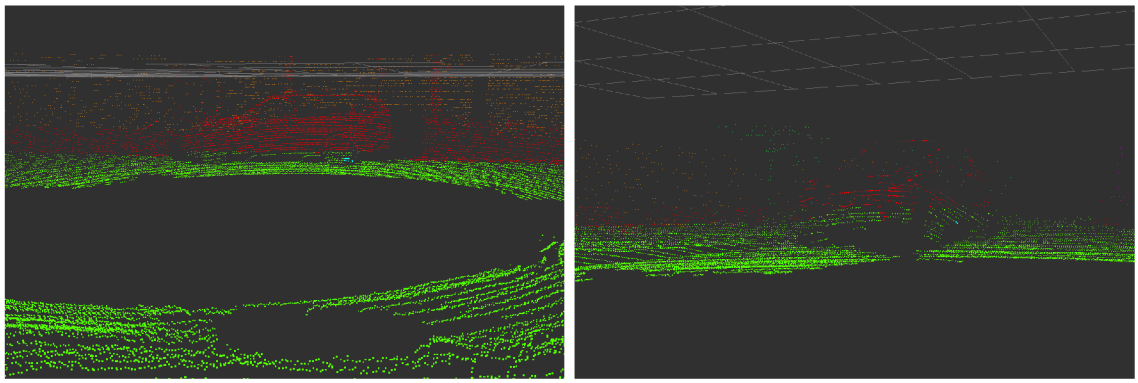
Príprava dát a programu

Pri prechode zo záznamov z Toad-a 2 som musel vyriešiť niekoľko väčších či menších prekážok. Prvou bolo pripravenie datasetu z RAW dát a spojiť ich s anotovanými dátami. RAW dáta boli uložené v binárnom súbore. Anotácie zase v matlabovom súbore.

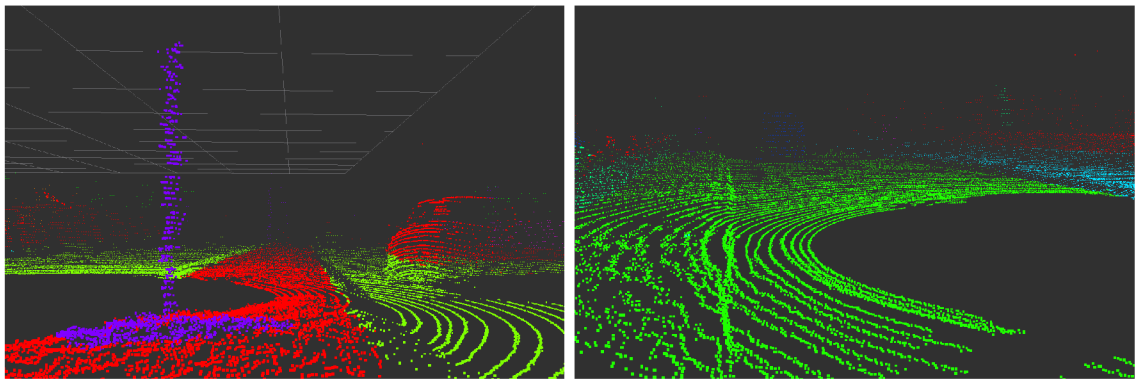
Velodyne data z RAW datasetov obsahoval tri súradnice bodu a jeho intenzitu/odrazivosť. Anotované dáta v tomto prípade predstavujú jednu novú celočíselnú hodnotu. Táto hodnota určuje do ktorej z desiatich kategorií bod patrí. Nie je ale oanotovaný každá snímka v zázname, ale každá desiata.



Obr. 8.5: Chyby v anotovanom datasete – zle anotované body.
Obrázok vľavom ukazuje zle anotované časti pred prednou maskou automobilu a po pravej strane. Obrázok vpravo sa pozerá na rovnakú scénu ale z iného uhla. Ďalšou zaujímavosťou okrem zle anotovaných pásov je ostrovček so značkou. Značka je síce anotovaná správne, ale ostrovček už nie.



Obr. 8.6: Chyby v anotovanom datasete.
Obrázky ukazujú zle anotované auto. Horná časť je braná ako prekážka, ale spodná časť už je opäť cesta.



Obr. 8.7: Chyby v anotovanom datasete – chodník označený ako cesta.
Ľavý obrázok ukazuje síce dobre anotovaný vyvýšený pás medzi pruhmy, ale nábežná hrana je ešte cesta. Obrázok vpravo ukazuje obrubník a časť chodníka ktoré sú zle anotované.

Pre uloženie a ďalšie spracovanie som použil textový formát PCD[2]. Spojenie a uloženie je naimplementované v jazyku GNU Octave. Z tohto formátu už nebol problém pomocou ROS balíčku *pcl_ros* vytvoriť BAG[3].

Korekcia výstupného obrázku o tieň automobilu a inými senzormi (kamery, GPS systém, samotný automobil), pre presnejšie meranie bola vytvorená mapa tienov.

Pre testovanie bolo potrebné previesť výslednú mapu na obrázok. Na vytvorenie obrázku som použil knižnicu OpenCV. Na získanie obrázkov z anotovaných dát som upravil pôvodný program tak, aby načítal mračno bodov z bag súboru, vytvoril mapu podľa anotáci a túto mapu následne uložil ako obrázok.

8.2 Testovanie

V tejto časti sú priblížené vyhodnocované mračná bodov – snímky, ukazatele, podľa ktorých sú vyhodnotené a jednotlivé merania aj s výsledkami.

Kvôli chybám v anotovanom datasete nie je možné overiť celú dĺžku záznamu. Z tohto dôvodu je vyhodnotených len pár zaujímavých snímkov. Tieto snímky sú podrobne popísané v ďalšej v podkapitole.

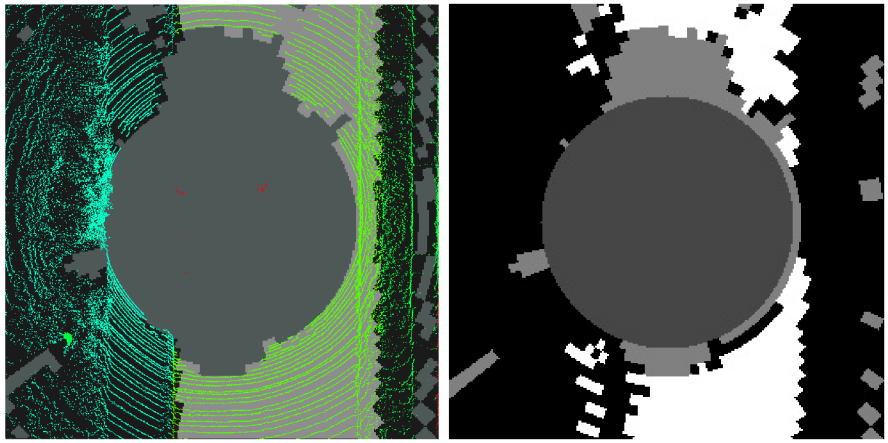
Pri vyhodnocaní jednotlivých snímkov sa nemenili vstupné prahy algoritmu, ciže všetky snímky z merania majú rovnaké vstupné prahy.

Pri vyhodnocovaní úspešnosti sú mapy zeme porovnávané pixel po pixeli, tiene auta a kruh v strede nie je zarátaný do úspešnosti detekcie. Toto porovnávanie je implementované ako jednoduchý skript napísany v programovacom jazyku Python.

Výber zaujímavých snímkov

Keďže nebolo možné vyhodnotiť dataset ako celok, rozhodol som sa, že pre testovanie použijem niečím zaujímavé snímky z rôznych datasetov.

Hlavným dôvodom pre výber snímok sú práve chyby v datasete. Dobrým príkladom je obrázok 8.8. Na snímke je vidieť takmer totožný obrázok. Ľavý je mračno bodov, ktoré je podfarbené anotovanou mapou zeme. Pravý obrázok je výsledok SVM s postupným učením. Úspešnosť detekcie cesty (biela farba) je len 55%.



Obr. 8.8: Porovnanie chyby anotácie snímku a výsledku online SVM

[2]Point Cloud Data http://pointclouds.org/documentation/tutorials/pcd_file_format.php
[3]http://wiki.ros.org/Bags/Format

Na obrázkoch (napr. 8.9 vľavo) je vidieť pohľad z vrchu (*topdown ortho*) na analyzovaný záber, ktorý je podfarbený anotovanou mapou. V strede obrázka je laserový senzor LiDAR. Svetlo zelené body sú cesta. Šedý kruh v strede ukazuje už neanalyzovanú časť pod minimálnou vzdialenosťou. Šedé pásy mimo stredový kruk sú tiene prednej kapoty (dole), kufra (hore) a iné senzori na streche auta. Všetky ostatné farby sú prekážka.

KITTI datasety obsahujú ako záznamy z laserového radaru LiDAR, tak aj záznam z predných kamier. Na obrázkoch (napr. 8.9 vpravo) je vidieť práve tieto fotky z prednej kamery. Každá z fotiek sa pozerá na jednu z vyhodnocovacích scén, ktoré môžme vidieť z pohľadu radaru na obrázkoch.

Prvá snímka (obrázok 8.9) ukazuje dvojprúdovú cestu. Na jednom okraji tráva na druhom obrubník a tesne za ním múrik. Zaujímavé na tejto snímke je, že trávnik je približne v rovnakej výške ako vozovka.

Obr. 8.9: Testovacia snímka 1

Na druhej snímke (obrázok 8.10) je vidieť priechod vozidla cez svetelnú križovatku s cyklistickým chodníkom. Aj ked oba strany priechodu sú v rovnakej výške, tak ten bližšie k železničnej trati je označený ako prekážka, pri čom ten na druhej strane zase ako voľná cesta. Na tesnom okraji analyzovaného mračna bodov je vidieť protiidúce auto. Zaujímave na tejto snímke je kolmá hrana vozidlo.

Obr. 8.10: Testovacia snímka 2

Tretia snímka (obrázok 8.11) ukazuje prejazd tenkou uličkou. Z jednej strany ohraničený dvoma stojácimi autami, zo strany druhej zase chodník s nízkym obrubníkom a stĺpiky. Strana so stĺpikmi je križovatka bočnej cesty.

Štvrtý snímok (obrázok 8.12) ukazuje rovnú trojprúdovu cestu ohraničenú vysokými obrubníkmi.

Na piatom snímku (obrázok 8.13) je vidieť opäť križovatku. Zaujímavá je veľmi nízkym obrubníkom v tesnej blízkosti auta a uhlom pod ktorým je druhá strana vozovky ohraničená

Obr. 8.11: Testovacia snímka 3

Obr. 8.12: Testovacia snímka 4

vysokým obrubníkom. Na anotovnom snímku je ale vidieť, že vysoký obrubník nie je po celej dĺže správne označený.

Obr. 8.13: Testovacia snímka 5

Šiesty snímok (obrázok 8.14) je dvojprúdová cesta s ohraničená na jednej strane obrubníkom a chodníkom. Na druhej strane zle anotovaným ostrovčekom oddelujúcim vozovku od železničnej trate.

Úspešnosť základného riešenia ground map

Aby som mohol vyhodnotit či úspešnosť SVM s postupným učením je vyššia než pôvodné riešenie, tak som musel toto riešenie zmerať. Výsledky merania sú v tabuľkách 8.1 a 8.2.

Parametre algoritmu:

maximálna nepravidelnosť cesty	0.04 m
maximálny rozdiel výšky	0.025 m
pravdepodobnosť cesty	0.9
pravdepodobnosť prekážky	0.5

Obr. 8.14: Testovacia snímka 6

Tieto parametre boli odhadnuté na základe experimentovania a vyhodnocovania na rôznych datasetoch.

Na prahovaných snímkoch je vidieť, že hrany prekážok sú vo väčšine prípadov rozpoznané. Opäť je problém s nepresnými snímkami v okolí auta.

Presnosť hľadania cesty na skúmaných snímkoch dosahuje maximálne 76%, minimálne 39% a priemer je 58%.

Úspešnosť SVM s postupným učením

Pri postupnom učení som najprv naučil SVM na iných datasetoch a až potom vyhodnotil chcenú snímku. Výsledky merania sú v tabuľkách 8.1 a 8.2.

Parametre algoritmu:

maximálna nepravidelnosť cesty	0.04 m
maximálny rozdiel výšky	0.025 m
pravdepodobnosť cesty	0.95
pravdepodobnosť prekážky	0.2

Na všetkých výsledných obrázkoch je vidieť, že body v najbližsom okolí auta sú veľmi často brané ako prekážka. Ako je vidieť aj na obrázkoch 8.2, tieto body sú nepravideľné ako v smere prstencov, tak aj vo výške nad zemou.

Presnosť hľadania cesty na skúmaných snímkoch dosahuje maximálne 87%, minimálne 55% a priemer je 71%.

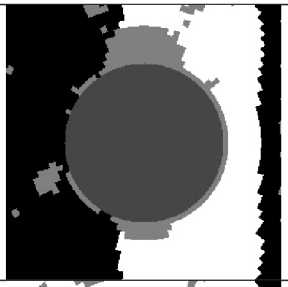
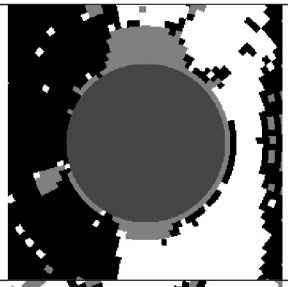
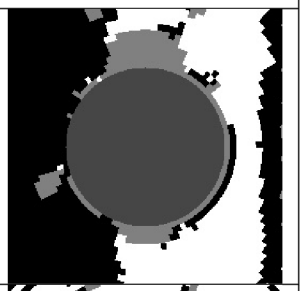
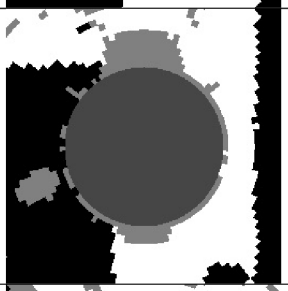
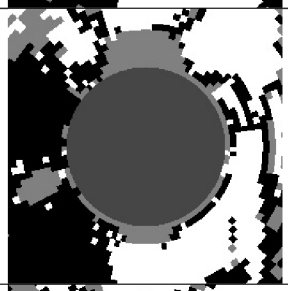
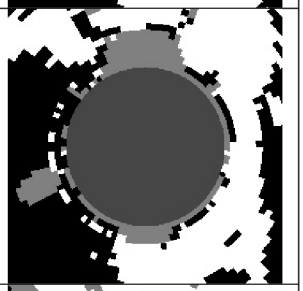
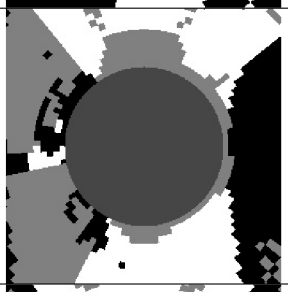
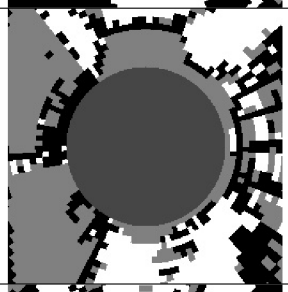
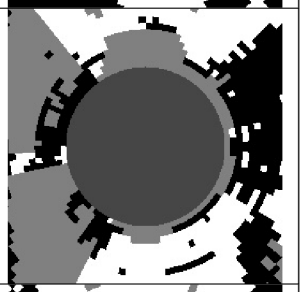
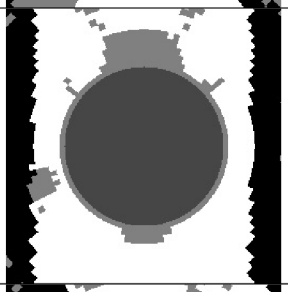
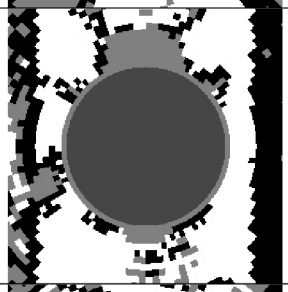
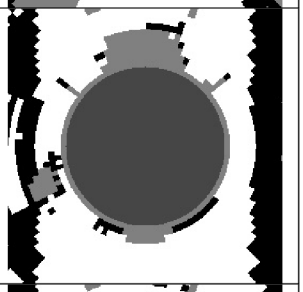
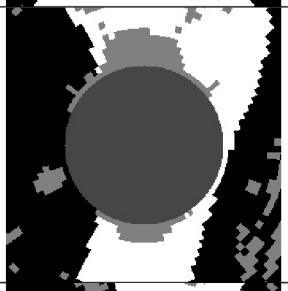
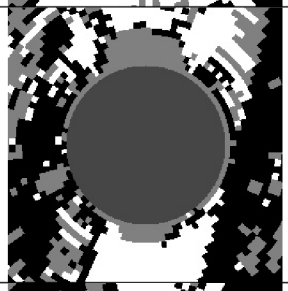
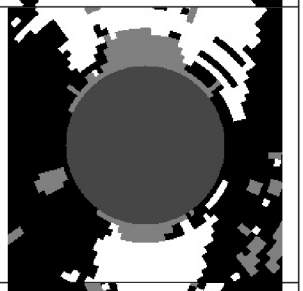
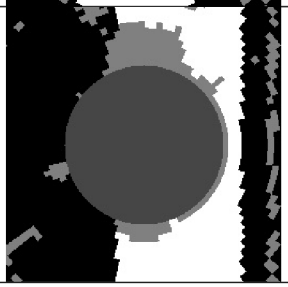
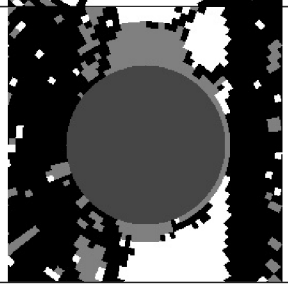
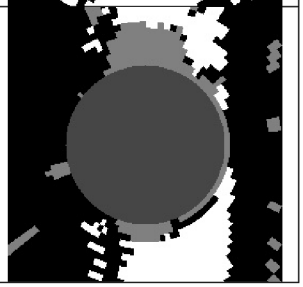
8.3 Vyhodnotenie a návrh pokračovania práce

Dataset KITTI som vybral kvôli jeho rozšírenosti.

Počas testovania a hlavne na výslednej snímke (tabuľka 8.1 snímka 2) je vidieť, že v tesnom okolí automobilu a senzorov sa často vyskytujú bunky ktoré sú označené ako prekážka. Tieto sektori sú označené svojím spôsobom správne, keďže dáta sú velmi blízko tieňov automobilu vo všetkých osách nerovné.

Algoritmus nebol navrhnutý tak, aby detekoval cestu, ciže zem s nejakou toleranciou pre náklon. Algoritmus bol navrhnutý tak aby hladal rovné plochy po ktorých by bolo možné ísť, aj keby to mal byť chodník. Čo často viedlo k tomu, že obrubník bol vyhodnotený ako prekážka, ale chodník už ako cesta. Ak by sa ale podarilo rospoznať obrubník bolo by možné nezaradovať tieto bunky do učenia. Hľadanie obrubníkov je nejednoznačná úloha, keďže hlavným faktorom určovania prekázok je výška nad zemou. Prvým problémom je

Tabuľka 8.1: Porovnanie anotovanej mapy s výsledkami ground_map a online SVM

Snímka	Anotácia	Ground_map	Online SVM
1			
2			
3			
4			
5			
6			

Tabuľka 8.2: Porovnanie výsledkov ground_map a online SVM

Snímka	Ukazateľ	Ground_map	Online SVM
1	Precision	0.7561	0.8728
	Recall	0.7692	0.8328
	True negative rate	0.8166	0.9120
	Accuracy	0.7965	0.8787
	F-measure	**0.7626**	**0.8524**
2	Precision	0.6124	0.6934
	Recall	0.6148	0.7521
	True negative rate	0.6283	0.6845
	Accuracy	0.6217	0.7175
	F-measure	**0.6136**	**0.7216**
3	Precision	0.5638	0.7262
	Recall	0.5197	0.6623
	True negative rate	0.3966	0.6533
	Accuracy	0.4705	0.6585
	F-measure	**0.5409**	**0.6928**
4	Precision	0.7975	0.8837
	Recall	0.6771	0.8586
	True negative rate	0.6359	0.7744
	Accuracy	0.6639	0.8305
	F-measure	**0.7324**	**0.8710**
5	Precision	0.3807	0.6153
	Recall	0.4033	0.5152
	True negative rate	0.5488	0.7974
	Accuracy	0.4896	0.6885
	F-measure	**0.3917**	**0.5608**
6	Precision	0.5327	0.7362
	Recall	0.3937	0.4451
	True negative rate	0.7753	0.9025
	Accuracy	0.6250	0.7290
	F-measure	**0.4528**	**0.5548**

samotný náklon cesty, druhým a rovnako zložitým je nedokonalosť cesty. Riešenie tohto problému by mohla vyriešiť fúzia dát s iným senzorom – napr. kamerou.

Na základe výsledkov merania sa Pegasonov algoritmus s príznakmi odvodenými od výšky bodu od zeme nehodí na jednoznačné určenie cesty v urbanistického prostredí.

Rozšírením práce by mohlo byť zrýchlenie spracovania dát pre SVM. Pri učení SVM bunku po bunke je potrebné nájsť a pristúpiť k celému jej okoliu. Toto by sa dalo jednoducho zoptimalizovať. Optimalizácia by spočívala len v predávaní už načítaných a stále potrebných buniek medzi iteráciami, čím by sa dosiahol menší počet čítaní z pamäťa.

Iným smerom ako rožíriť túto prácu by mohlo byť automatické hľadanie prahov a parametrov pre SVM. Keďže v implementácií sú použité štyri prahy pre hľadanie extrémných snímkov a SVM, kde sú použité dva parametre, tak by za zváženie stálo použiť openMPI.

Ako bolo už spomínané vyššie v urbanistickom prostredí by bolo použiť kameru ako daľší zdroj informácie. Kameru použiť napríklad na hľadanie obrubníkov.

Kapitola 9

Záver

Cielom tejto práce bolo získať povedomie o súčastných metódach ako vyhodnotiť okolie autonómneho vozidla za pomoci laserového radaru Velodyne LiDAR a kamery. Vybral som si jedno z existujúcich riešení a rozšíril som ho. Pôvodné riešenie používalo k rozhodnutiu či je zem v okolí cesta prahy. Tieto prahy som nahradil za strojové učenie SVM s postupným učením.

Porovnanie týchto prístupov som spravil nad datasetom KITTI, ktorý obsahuje aj anotované dáta z Velodyne LiDAR. Pre porovnanie som použil ukazateľ F-measure.

Anotácie v datasete KITTI neboli dotiahnuté do úplnej dokonalosti, tak nebolo možné merať celé datasety. Preto som zvolil len určité snímky.

Pôvodný algoritmus dosahuje priemernú presnosť odhadu cesty 58%. Moja implementácia 71%. Hlavným nedostatkom povôdného riešenia je veľký počet neodhadnutých buniek. Nevýhodnou nového riešenia je rýchlosť.

Nový algoritmus bol navrhnutý tak aby hladal rovné plochy po ktorých by bolo možné ísť, aj keby to mal byť chodník. Čo často viedlo k tomu, že obrubník bol vyhodnotený ako prekážka, ale chodník už ako cesta. Pokračovaním tejto práce by mohlo byť fúzia mračna bodov z Velodyne LiDAR s kamerou. Kamera by mohla hladať hrany obrubníkov a tým zamedzila odhadu rovných plôch za prekážkou. Ďalším vylepšením by mohla byť optimalizácia opätovného prechodu buniek.

Literatúra

[1] but_velodyne_proc. [online], 2016.
URL http://wiki.ros.org/but_velodyne_proc

[2] HDL-32E. [online], 2016.
URL http://velodynelidar.com/docs/datasheet/97-0038_Rev%20E_%20HDL-32E_Datasheet_Web.pdf

[3] Point Cloud Library. [online], 2016.
URL http://pointclouds.org/

[4] ROS. [online], 2016.
URL http://www.ros.org/about-ros/

[5] ROS. [online], 2016.
URL http://wiki.ros.org/ROS/Introduction

[6] Výzkumná skupina robotiky Robo@FIT. [online], 2016.
URL http://www.fit.vutbr.cz/research/groups/robo/eqp.php.cs

[7] Wikipedia: Support vector machine. [online], 2016.
URL https://en.wikipedia.org/wiki/Support_vector_machine

[8] Christopher Rasmussen: RoadCompass: following rural roads with vision + ladar using vanishing point tracking. *Auton Robot*, ročník 25, 2008: s. 205–229.

[9] Geiger, A.; Lenz, P.; Stiller, C.; aj.: Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[10] J. Han, D. Kim, M. Lee and M. Sunwoo: Road boundary detection and tracking for structured and unstructured roads using a 2D Lidar sensor. *International Journal of Automotive Technology*, ročník 15, 2014: s. 611–623.

[11] King, D. E.: Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, ročník 10, 2009: s. 1755–1758.

[12] Martin Velas, Michal Spanel, Zdenek Materna, Adam Herout: Calibration of RGB Camera With Velodyne LiDAR. *WSCG 2014 Communication Papers Proceedings*, ročník 1, 2014: s. 135–144.

[13] Michal Spanel: Road Detection from Velodyne Lidar Data. 2013.

[14] Patric Y. Shinzato, Denis F. Wolf, Christoph Stiller: Road Terrain Detection: Avoiding Common Obstacle Detection Assumptions Using Sensor Fusion. *IEEE Intelligent Vehicles Symposium*, ročník 4, 2014: s. 687–692.

[15] Pavel Laskov, Christian Gehl, Stefan Kruger, Klaus-Robert Muller: Incremental Support Vector Learning: Analysis, Implementation and Applications. *Journal of Machine Learning Research*, ročník 7, 2006: s. 1909–1936.

[16] Richard Zhang, Stefan A. Candra, Kai Vetter, Avideh Zakhor: Sensor fusion for Semantic Segmentation of Urban Scenes. *ICRA*, 2015.

[17] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, Andrew Cotter: Pegasos: primal estimated sub-gradient solver for SVM. *Math. Program., Ser. B*, ročník 25, 2010: s. 3–30.

[18] Yuan Xia, Zhao Chun-Xia, Zhang Hao-Feng: Road Detection and Corner Extraction Using High Definition Lidar. *Information Technology Journal*, ročník 9, 2010: s. 1022–1030.

Prílohy