

Master Thesis



Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Change Detection in Vehicle Neighbourhood Using 3D Data and Aggregated 3D Map

Bc. Dominik Fiala

Supervisor: Ing. Martin Matoušek, Ph.D.

Field of study: Open Informatics

Subfield: Computer Vision and Image Processing

May 2018

I. Personal and study details

Student's name: **Fiala Dominik** Personal ID number: **406461**
Faculty / Institute: **Faculty of Electrical Engineering**
Department / Institute: **Department of Cybernetics**
Study program: **Open Informatics**
Branch of study: **Computer Vision and Image Processing**

II. Master's thesis details

Master's thesis title in English:

Change Detection in Vehicle Neighbourhood Using 3D Data and Aggregated 3D Map

Master's thesis title in Czech:

Detekce změn v okolí vozidla pomocí 3D dat a agregované 3D mapy

Guidelines:

1. Propose a representation of a 3D map suitable for detection and segmentation of scene changes.
2. Choose a method for acquisition and off-line aggregation of 3D measurements (points) into the map.
3. Develop a method for on-line segmentation of new 3D measurements into known (static) scene and new objects, using the map.
4. Develop an implementation of these algorithms and verify them on a real data provided by omnidirectional LiDARs mounted on a vehicle. The data will be provided.
5. Propose a series of experiments with objects of different sizes and speeds (static and slowly moving) and demonstrate the performance of the method.

Bibliography / sources:

- [1] Hornung, A., Wurm, K.M., Bennewitz, M. et al. - OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees - Autonomous Robots (2013) 34: 189.
[2] Saarinen, J., Andreasson, H., Stoyanov, T., Lilienthal, A. J. - 3D Normal Distributions Transform Occupancy Maps: An Efficient Representation for Mapping in Dynamic Environments - International Journal of Robotics Research (2013) 32: 14.

Name and workplace of master's thesis supervisor:

Ing. Martin Matoušek, Ph.D., Robotic Perception, CIIRC

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **11.01.2018** Deadline for master's thesis submission: **25.05.2018**

Assignment valid until: **30.09.2019**

Ing. Martin Matoušek, Ph.D.
Supervisor's signature

doc. Ing. Tomáš Svoboda, Ph.D.
Head of department's signature

prof. Ing. Pavel Ripka, CSc.
Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt

Student's signature

Acknowledgements

I would like to thank the supervisor of the thesis, Ing. Martin Matoušek, Ph.D. for his support, patience and time which he dedicated to me while working on this thesis. Also, I would like to thank to my family and friends who have been a great support throughout the whole process.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instruction for observing the ethical principles in the preparation of university theses.

Prague, May 25, 2018

Abstract

Novel approach for detecting changes in neighbourhood of moving autonomous vehicles is presented in this thesis. The work is based on a 3D map constructed from 3D measurements provided by set of LiDARs mounted on a vehicle. The changes are detected by comparing the map with current 3D data.

To obtain suitable representation of 3D scene, we propose Normal Distribution Transform Belief Map (NDT-BM) which combines known Normal Distribution Transform Occupancy Map (NDT-OM) and Transferable Belief Model (TBM). From NDT-OM we use the grid structure and the ability to preserve information about mass distribution within each cell. TBM is used for better modeling dynamic behavior of urban environment. To detect changes, we propose segmentation process based on computation of distance between elements of NDT-BM and new 3D measurements. To handle a noise in LiDAR's measurements, we established heuristics based on clustering.

In real-world experiments we demonstrated the ability of NDT-BM to correctly map static objects and cope with the dynamic ones. We have shown that we are able to detect static and dynamic changes of various speed and size too.

Keywords: 3D data aggregation, change detection, 3D map, LiDAR, autonomous car

Abstrakt

V této práci je prezentován nový přístup k detekci změn v okolí jedoucích autonomních vozidel. Práce jsou založeny na 3D mapě vytvořené z 3D měření poskytovaných sadou LiDAR sensorů namontovaných na vozidle. Změny jsou detekovány porovnáním mapy s aktuálními 3D daty.

Pro získání vhodné reprezentace 3D scény, navrhujeme tzv. Normal Distribution Transform Belief Mapu (NDT-BM), která kombinuje již existující tzv. Normal Distribution Transform Occupancy Mapu (NDT-OM) a tzv. Transferable Belief Model (TBM). Z NDT-OM používáme strukturu mřížky a schopnost uchovat informace o masové distribuci v každé buňce. TBM je použit pro lepší modelování dynamického chování městského prostředí. Pro detekci změn navrhujeme segmentační proces založený na výpočtu vzdálenosti mezi prvky NDT-BM a novými 3D měřeními. K řešení problémů se šumem jsme použili heuristický přístup založený na shlukování.

V reálných experimentech jsme prokázali schopnost NDT-BM správně mapovat statické objekty a vypořádat se s těmi dynamickými. Ukázali jsme, že dokážeme detekovat statické i dynamické změny, a to o různých rychlostí a velikostí.

Klíčová slova: agregace 3D dat, detekce změn, 3D mapa, LiDAR, autonomní automobil

Contents


1 Introduction	1	6 Conclusions	41
2 Related work	3	6.1 Summary	41
3 Aggregation of 3D data	5	6.2 Limitations and open problems .	42
3.1 Evidential grid mapping	6	6.3 Future work	42
3.1.1 Combination of evidence	7	Bibliography	45
3.1.2 Belief mass derivation	7	A Contents of attached CD	47
3.1.3 Handling of conflict	11		
3.2 Normal Distribution Transform (NDT) model	11		
3.2.1 Computation of NDT parameters	11		
3.2.2 Computation of inverse covariance matrix	12		
3.2.3 Relative distance between ray and mass	12		
3.3 Implementation details	13		
3.3.1 Node initialization	14		
3.3.2 Workflow	14		
4 Change detection in vehicle neighborhood	17		
4.1 3D map reduction	17		
4.2 Points segmentation	18		
4.3 Change clustering and tracking .	19		
4.4 Implementation details	20		
5 Experiments	23		
5.1 Description of used datasets	23		
5.1.1 KITTI	23		
5.1.2 NCLT	24		
5.2 Parameters setup	24		
5.3 Proposed methods evaluation . .	24		
5.3.1 Test scenario 1 (TS1)	26		
5.3.2 Test scenario 2 (TS2)	26		
5.3.3 Test scenario 3 (TS3)	28		
5.3.4 Test scenario 4 (TS4)	32		
5.4 Comparing ALG-AGG with OctoMap framework	32		

Figures

3.1 Overall workflow of 3D data aggregation	6	5.14 Occupancy map created by OctoMap from KITTI-18	39
3.2 The dependence of basic belief masses on ρ	9		
3.3 Evolution of voxel's basic belief masses	10		
3.4 An illustration of relative distance between ray and mass	13		
3.5 An illustration of aggregation process	15		
3.6 Flowchart of node's state updating process	15		
4.1 Overall workflow of change detection process	18		
4.2 An illustration of points segmentation process	19		
5.1 Visualization of the input point cloud for KITTI-18	26		
5.2 NDT-BM for recording KITTI-18	27		
5.3 Detail of the 3D map	28		
5.4 Visualization of the input point cloud for NCLT-02-P01, NCLT-04-P01, NCLT-06-P01	29		
5.5 Evolution of NDT-BM	30		
5.6 NDT-BM generated from multiple recordings	31		
5.7 Change detection of new static objects	33		
5.8 Visualization of the input point cloud for KITTI-17	34		
5.9 Moving objects segmentation	35		
5.10 Moving objects clustering and tracking	36		
5.11 Visualization of the input point cloud for NCLT-09-P02	37		
5.12 Segmentation of cyclist	37		
5.13 Clustering and tracking of cyclist	38		

Tables

3.1 List of basic belief masses of B_{hit} (second column) and B_{miss} (third column)	8
3.2 List of additional properties of NDT-BM's node	14
5.1 Selected NCLT recordings	24
5.2 List of all used parameters	25



Chapter 1

Introduction

Our aim of this thesis is to propose method which will help autonomous vehicles to detect changes in its neighborhood during ride in urban environment. One motivation for this work is to improve safety of autonomous driving. According to WHO's statistic, in 2013 there were 1.25 million road deaths around the world and they predict that road traffic injuries become the seventh leading cause of death by 2030. Second motivation aims to increase comfort for passengers. Consider the scenario of autonomously finding parking spot in a busy city, while user need not be inside the vehicle.

All of mentioned reasons have one common attribute which autonomous cars need to have. They should have knowledge about urban environment in advance, to be able to plan paths, to accurately predict behavior of other subjects on the road etc. Since the urban environment is dynamic and from day-to-day new wall by the road can be built, smart car needs to be able to detect these changes as well.

Hence, we focus in this work on what are the options to represent 3D scene, how can be improved and how to use it to detect changes.

Following this introduction, in chapter 2 related works in the field of 3D mapping are studied and compared. In the next following two chapters, chapter 3 and chapter 4, we presents our approach of 3D mapping and change detection. In chapter 5 set of experiments is described and analyzed. At the end, in chapter 6, the thesis is summarized.



Chapter 2

Related work

One of the widely used representations of 2D and 3D space across various robotics tasks, are Occupancy grid maps [Elf89]. An Occupancy map represents the environment as a set of cells, created by uniform space partitioning. The binary label is assigned to each of those cells, to distinguish if it contains objects (cell is *occupied*) or not (cell is *free*). Classification process is based on thresholding of *occupancy* – probability of being occupied. Occupancy for each cell is, independently to each other, updated with every new observation about its current state. As observations, the measurements from sonar or LiDAR sensor are usually used. Then the updating step for specific cell is done by increasing (reflected point lies within the cell's boundaries) or decreasing (cell lies between sensor and reflected point) its occupancy by some constant factor.

There are two assumptions in the occupancy grid map model, which can cause inaccurate results. Firstly, it is assumed that the space is uniform within each cell. Secondly, it is assumed that the environment consists of static objects, so it is not explicitly trying to model the dynamics [SASL13]. The first assumption causes problems when the size of the grid cell is bigger than an object, so the object occupies only the part of the cell's space and LiDAR's rays or sonar's waves can go through this cell, if they are in appropriate position. This induces contradictory observations, which can lead to wrong classification (e.g., cell with mass is labeled as free). The shortcomings of the second assumption emerge when the map is used in a dynamic environment (e.g., urban environment). Depending on the mutual position of a moving object and a sensor, a moving object can leave a *trace*, a set of cells incorrectly labeled as occupied, in the map.

A simple way to compensate for the drawback of the first assumption, is to set the size of a cell to be lower than the size of the smallest objects which can

occur in the mapped environment. But intuitively, with decreasing cell's size, the memory consumption of the whole map is growing.

Authors in [SASL13] overcome both problems by using *Normal Distribution Transform* (NDT) representation [BS03] to model, how points are generated from a surface within a cell. In proposed 3D representation for mapping called Normal Distribution Transform Occupancy Map (NDT-OM) they enhance each cell of occupancy grid with the NDT element, to be able to track the distribution of a mass within the cell. In the update step they adjust the value of the increment according to position of the cell's distribution and processed observation.

In the article [PNDW98], authors discuss another inaccuracy in occupancy grid maps. In occupancy grid map by [Elf89], value of occupancy after initialization is 0.5. Thus, after cell's initialization we can say there is 50% chance that it is occupied, even we have not received any information yet. In addition, they mention if the occupancy of cell is close to 0.5, no one can determine if it is because contradictory observations were obtained for that cell (related to mentioned problems above), or that not enough information has been received. To cope with this problem, they use so-called Evidential approach. They represent the environment as a grid with uniformly splitted cells, but instead of Bayesian probabilistic method for fusion of sensor's information, they use the Dempster-Shafer theory of evidence (DST) [Sha76]. The advance of DST is, that it allows to explicitly model these two situations, by proper definition of a frame of discernment. In the articles [TTWB14, TW17], authors go further and using DST they create the environment model capable to represent free space, static occupancy and dynamic occupancy, which they use for tracking and mapping in dynamic urban environment.

There are many others way how to represent 3D scene. For example, modeling objects in scene from 3D point cloud can be done by polygonal meshes [SYM10]. This approach leads to very precise representation, but at the cost of high memory consumption. In [PGK02] they introduce methods for simplification of input 3D point cloud which can be used to compress amount of data points before the polygonal mesh is created. In [SLB18], authors novel mapping technique is based on representing a map not in the position domain (as the all previously mentioned approaches), but in the discrete frequency domain and using inverse discrete cosine transform to convert it to a continuously differentiable scalar field in the position domain.

In this work, we propose a new approach to create 3D representation of dynamic environment, based on combination of NDT, NDT-OM and TBM models.

Chapter 3

Aggregation of 3D data

Before we describe the details, we would like to introduce the main idea and flow of the data aggregation process (see Figure 3.1), whose output is a *3D map* – representation of 3D scene.

We have a vehicle equipped with LiDAR sensor and navigation platform. LiDAR provides 3D measurements (i.e., 3D points of reflected obstacles) and navigation platform serves as source of vehicle's and LiDAR's current position in global coordinate system. Then, we make a ride with this vehicle and obtain 3D measurements of appropriate 3D scene. After that, acquired raw 3D points are transformed from sensor's coordinate system into global coordinate system using navigation platform by process called *registration*. These transformed 3D points, denoted as *point cloud*, are sent to the server. When the server obtains sufficient amount of point clouds of particular 3D scene or if it already has a 3D map for this scene (from previous run of data aggregation process), the 3D map is created or updated, respectively, by proposed 3D data aggregation algorithm (ALG-AGG). Then this map will be used to determine what has changed in vehicle's surrounding (see chapter 4).

In this work, the 3D map is a 3D grid dividing 3D scene into equally large cells. Every such cell (called *voxel*) can keep any information about the space (as voxel's property) what it limits with its borders. The basic property in our 3D map is label if the voxel is *free* (voxel does not contain any amount of static mass), *occupied* (voxel contains some amount of static mass) or *dynamic occupied* (free voxel which is occasionally temporarily occupied).

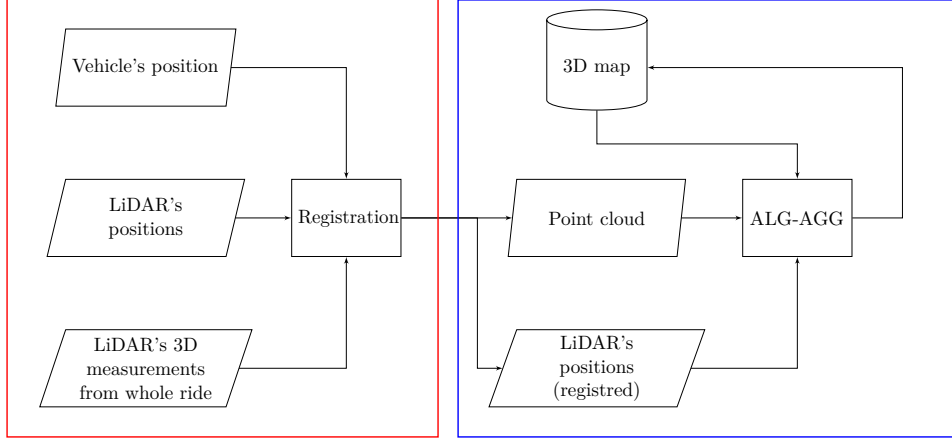


Figure 3.1: Overall workflow of 3D data aggregation process. At first, all LiDAR’s 3D measurements and positions from whole ride are registered into global coordinate system. Then algorithm ALG-AGG is used and 3D map is created or updated. Red and blue frame denote the part of the process running on the side of the vehicle and server, respectively.

3.1 Evidential grid mapping

Common approach to determine which voxels are free or occupied is to use occupancy grid [Elf89, SASL13], where for each voxel the *occupancy* is calculated and then the labeling is done by thresholding.

In our work we use extension to this approach. Inspired by [MCB11, TTWB14, TW17], we decided to use *The Transferable Belief Model* (TBM) [Sme94]. This framework should help us to overcome inaccuracies caused by dynamic objects when occupancy grid mapping is used.

By system S in this task we consider set of voxels, the frame of discernment is

$$\Omega = \{F, O, D\}, \quad (3.1)$$

representing hypotheses that space is free, occupied or dynamic occupied. The power set of Ω is

$$2^\Omega = \{\emptyset, F, O, D, \{O, D\}, \{F, O\}, \{F, D\}, \Omega = U\}, \quad (3.2)$$

where U represents *unknown* state and \emptyset represents *conflict* which indicate that for the appropriate voxel we observed two or more contradictory evidences (e.g., for voxel which we consider as a free, we observe evidence indicating it is occupied).

In this work, set of all basic belief masses for all propositions from 2^Ω will be denoted as B and their sum will always be 1,

$$\sum_{A \in 2^\Omega} m(A) = 1. \quad (3.3)$$

3.1.1 Combination of evidence

To determine the correct state of our system S , we do some independent measurements by various sensors. Every such information is in TBM called as *evidence* and it can be represented by B .

We assume that LiDAR sensor's with known position during measurement are used as the source of information about S . Then we can count with two types of evidences assignable to any voxel v :

- *hit evidence* B_{hit} corresponding to the position of the reflected point which is inside v ,
- *miss evidence* B_{miss} corresponding to the part of the laser ray which goes completely through v .

We will use term *evidence integration* to denote combination of evidence's basic belief masses with voxel's basic belief masses using conjunctive combination rule [Sme07],

$$m_{12}(A) = \sum_{X \cap Y = A} m_1(X) m_2(Y), \quad \forall A \subseteq \Omega. \quad (3.4)$$

3.1.2 Belief mass derivation

We added two counters, α and β , as another voxel's properties, to be able to track number of B_{hit} and B_{miss} evidences integrated into voxel and to compute ratio $\rho = \frac{\alpha}{\alpha + \beta}$. Value range of $\rho \in \langle 0, 1 \rangle$ can be split into 3 parts which have different effect on evidence integration:

- $\langle 0, \rho_L \rangle$ – more B_{miss} than B_{hit} have been integrated into voxel,
- $\langle \rho_L, \rho_U \rangle$ – the count of integrated evidences is roughly similar for both ev. types,
- $\langle \rho_U, 1 \rangle$ – more B_{hit} than B_{miss} have been integrated into voxel,

where $\rho_L, \rho_U \in \mathbb{R}$, $0 < \rho_L < \rho_U < 1$.

If B_{miss} should be integrated into voxel with $\rho \in \langle 0, \rho_L \rangle$, we can assume the credibility and belief that voxel is free to be growing as the value of ρ is getting closer to 0. Hence, with decreasing ρ , we increase value of $m(O)_{\text{hit}}$. Similarly, if B_{hit} should be integrated into voxel with $\langle \rho_U, 1 \rangle$, we can assume the credibility and belief that voxel is occupied to be growing as the value of ρ is getting closer to 1. Hence, with increasing ρ , we increase value of $m(F)_{\text{hit}}$.

If $\rho \in \langle \rho_L, \rho_U \rangle$, it indicates that the part of the environment voxel it represents is dynamic. Therefore, we increase $m(D)$, $m(O, D)$ for integrated evidence with increasing ρ until $\rho = \frac{\rho_L + \rho_U}{2}$, then we start decrease their values.

If $\rho \in \langle \rho_U, 1 \rangle$ or $\langle 0, \rho_L \rangle$ and contrary evidence B_{miss} or B_{hit} , respectively, should be integrated into voxel, it either pointing on change in the environment (e.g., new building has been built), which has not been registered yet. Or this new evidence is outlier (due to some noise in LiDAR measurements). Thus,

$A \in 2^\Omega$	$m(A)_{\text{hit}}$	$m(A)_{\text{miss}}$
\emptyset	0	0
F	0	$g(\rho, \lambda_F, \mu_F = 0, \sigma_F)$
O	$g(\rho, \lambda_O, \mu_O = 1, \sigma_O)$	0
D	$g(\rho, \lambda_D^{\text{hit}}, \mu_D, \sigma_D^{\text{hit}})$	$g(\rho, \lambda_D^{\text{miss}}, \mu_D, \sigma_D^{\text{miss}})$
$\{O, D\}$	$g(\rho, \lambda_{OD}^{\text{hit}}, \mu_{OD}, \sigma_{OD}^{\text{hit}})$	$g(\rho, \lambda_{OD}^{\text{miss}}, \mu_{OD}, \sigma_{OD}^{\text{miss}})$
$\{F, O\}$	0	0
$\{F, D\}$	0	0
U	$1 - \sum_{A \in 2^\Omega \setminus U} m(A)_{\text{hit}}$	$1 - \sum_{A \in 2^\Omega \setminus U} m(A)_{\text{miss}}$

Table 3.1: List of basic belief masses of B_{hit} (second column) and B_{miss} (third column)

as ρ is approaching to 1 or 0, we increase $m(U)_{\text{miss}}$ or $m(U)_{\text{hit}}$, respectively (i.e., we decrease credibility of evidence).

To model this proposed behavior, we use Gaussian function for each of basic belief mass,

$$g(\rho, \lambda, \mu, \sigma) = \lambda \cdot \exp\left(-\frac{1}{2} \frac{\rho - \mu}{\sigma}\right), \quad (3.5)$$

where ρ is its input variable and other three inputs are parameters defining a shape of Gaussian curve (λ – peak’s height, μ – position of the peak’s center, σ – curve’s width).

Based on text above, values of basic belief masses of B_{hit} and B_{miss} are listed in Table 3.1. An example how values of B_{hit} and B_{miss} changed with respect to ρ can be seen in Figure 3.2. In Figure 3.3 we show the evolution of voxel’s basic belief masses when different evidences are integrated in it.

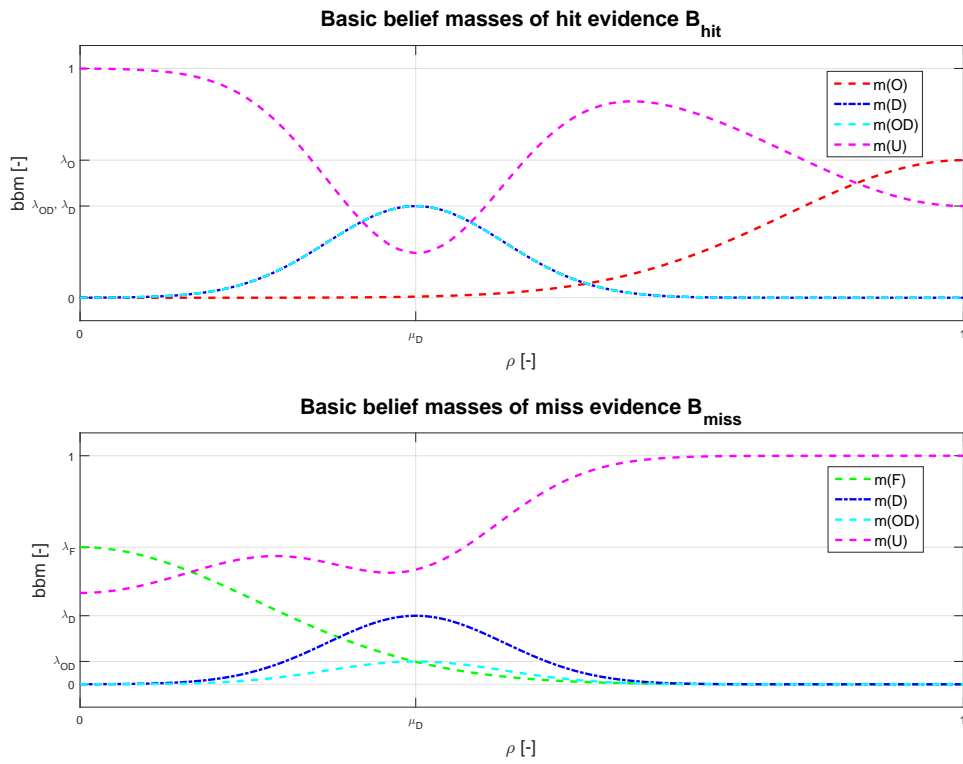


Figure 3.2: The dependence of basic belief masses on ρ . Only the non-zero bbms are shown.

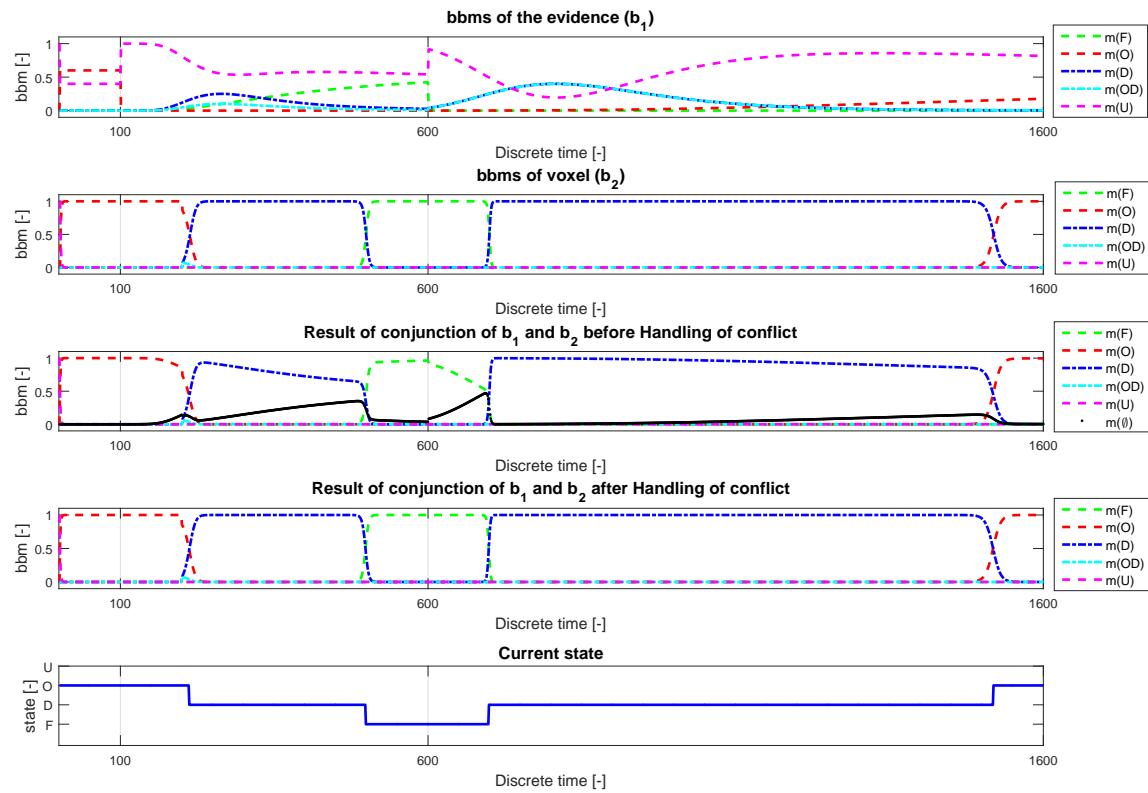


Figure 3.3: Evolution of voxel's basic belief masses using different input evidences. In the first four graph, only the non-zero bbm are shown.

3.1.3 Handling of conflict

Urban environment is a dynamic one and for that reason, a lot of *conflict* situations can arise during the aggregation process. As was mentioned in [Sme07], the way how the conflict can be handled strongly depends on the context of the task. Therefore there is no general approach how to do it and we should design our own solution based on his experience and knowledge about appropriate system S .

When a conflict arises for voxel v_i , we check if current state s_i is D and if expression $B_L \leq \rho \leq B_U$ is true. If so, the value of $m(\emptyset)$ is equally divided between $m(D)$ and $m(\{O, D\})$. Otherwise we normalize all basic belief masses m in B_i ,

$$\forall A \subseteq \Omega, m(A) = \begin{cases} \frac{m(A)}{1-m(\emptyset)} & \text{if } A \neq \emptyset \\ 0 & \text{if } A = \emptyset \end{cases}. \quad (3.6)$$

3.2 Normal Distribution Transform (NDT) model

Regular partition of the space can create voxels which contain non-uniformly distributed mass. LiDAR's rays can go through these voxels, reporting incorrectly that they are free. To compensate this drawback, we keep the information about distribution of points in each voxel (in property N) and it is represented by normal distribution $\mathcal{N}(\mu, C)$ [Mag09, SASL13], which parameters are sample mean μ and sample covariance matrix C and they can be estimated from measured 3D points,

$$\mu_i = \frac{1}{n} \sum_{k=1}^n z_k, \quad (3.7)$$

$$C_i = \frac{1}{n-1} \sum_{k=1}^n (z_k - \mu)(z_k - \mu)^T, \quad (3.8)$$

where z_k is k -th point in the i -th voxel and n is total amount of the points in that voxel.

3.2.1 Computation of NDT parameters

If we worked in aggregation process directly with μ and C and we would like to update some 3D map, we would have to keep in memory all inserted points which would have negative impact on speed and memory performance. Therefore, we will use the Recursive Sample Covariance (RSC) method proposed in [SASL13].

If in the text a sample mean μ and a sample covariance matrix C is mentioned, then it means that it was derived from T and S , respectively.

3.2.2 Computation of inverse covariance matrix

In most cases, to compute inverse matrix C^{-1} for covariance matrix C , standard methods can be used. But there are two main problems which we need to handle. Firstly, in the case that points are coplanar or colinear, C is singular and then it is not possible to obtain C^{-1} . Secondly, due to problem with rounding when floating points are used in calculation on computers, it can be impossible to C^{-1} even for nearly singular matrix. To overcome these issues, we have common solution for both problems and it is based on eigenvalues decomposition mentioned in [Mag09]:

1. Test the ratio between the largest eigenvalue λ_3 of C and the other eigenvalues λ_1 and λ_2 .
2. If λ_1 is ω -times smaller than λ_3 , replace it with $\lambda'_1 = \frac{\lambda_3}{\omega}$. Do the same analogically for λ_2 .
3. After that, compose a new covariance matrix

$$C' = V\Lambda'V^{-1}, \quad (3.9)$$

where V contains eigenvectors of C and

$$\Lambda' = \begin{bmatrix} \lambda'_1 & 0 & 0 \\ 0 & \lambda'_2 & 0 \\ 0 & 0 & \lambda'_3 \end{bmatrix}. \quad (3.10)$$

4. Then compute C^{-1} using C' applying any standard method.

3.2.3 Relative distance between ray and mass

One of the advantages of using NDT model is the possibility to compute a relative distance between points located in particular voxel and ray (representing miss evidence) which goes through that voxel. This knowledge is used to determine if the ray interferes with the voxel's mass or not.

The first part of this procedure is to obtain point x_{ML} which lies on the ray within the voxel and have maximal value of likelihood $p(x_{ML}|N(\mu, C))$, where $N(\mu, C)$ represents normal distribution of points in the voxel. This is done by analytical solution in chapter 3.4.1 in [SASL13]. When we know the position of x_{ML} we compute its likelihood [SASL13],

$$p(x_{ML}|N(\mu, C)) \approx \exp(-0.5(x_{ML} - \mu)^T C^{-1}(x_{ML} - \mu)). \quad (3.11)$$

For our work, we define function $d_R(r, N)$ which computes *relative distance* between ray r and voxel's NDT property N by two-step procedure described above.

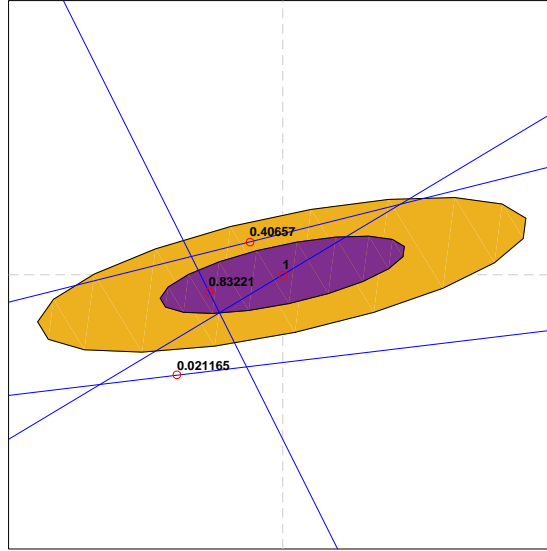


Figure 3.4: An illustration of relative distance between ray and mass. Rays are plotted in blue color and points with maximal value of likelihood $p(x_{ML}|N(\mu, C))$ are shown as red circle. 2-D Gaussian, representing the voxel’s mass, is plotted as set of ellipses with standard deviation 1 and 2.

3.3 Implementation details

For handling the division of the space into voxels we use octree data structure (based on the OctoMap library [HWB⁺13]), which every leaf node corresponds to one voxel. Besides the original properties from the OctoMap framework, we also store in each node of the octree additional properties which are listed in Table 3.2. Because basic belief masses for $\{F, O\}$ and $\{F, D\}$ are for both evidences zero (see subsection 3.1.2) and are zero after initialization (see subsection 3.3.1), too, their value will always be zero (see Equation 3.4). Therefore, we can omit them to decrease memory consumption.

We named this data structure as *Normal Distribution Transform Belief Map* (NDT-BM), which reflects 2 main ideas behind it – Normal Distribution Transform model and Transferable belief model.

One parameter of NDT-BM should be mentioned, namely octree’s resolution δ (i.e., length of voxel’s edge). The result and performance of 3D data aggregation is strongly dependent on the value of δ – with decreasing value of δ , the quality of result map will increase, but also the run-time and the memory consumption.

For simplification, to refer node’s properties, we will use directly the symbol of the property with the node’s index (e.g., for node n_i we will use $bbms_i, X_i, \dots$).

Property	Initial value	Description	
B	$m(\emptyset)$	0	bbm of null set
	$m(F)$	0	bbm of free proposition
	$m(O)$	0	bbm of occupied proposition
	$m(D)$	0	bbm of dyn. occupied proposition
	$m(\{O, D\})$	0	bbm of $\{O, D\}$ proposition
	$m(U)$	1	bbm of unknown proposition
P	$p(F)$	$\frac{1}{3}$	pignistic probability of F
	$p(O)$	$\frac{1}{3}$	pignistic probability of O
	$p(D)$	$\frac{1}{3}$	pignistic probability of D
N	n	0	counter of processed points
	X	\emptyset	set for storing unprocessed points
	T	null	vector for continuous update of sample mean μ
	S	null	matrix for continuous update of sample covariance matrix C
s	U	current node's state	
α	0	counter of hit evidences	
β	0	counter of miss evidences	

Table 3.2: List of additional properties of NDT-BM's node

3.3.1 Node initialization

Every time, when new NDT-BM's node is created, properties are initialized as it stated in Table 3.2. These values are based on fact, that in the time of initialization we have no evidences about voxel's state, so its state is unknown for us.

3.3.2 Workflow

The core part of this process is the proposed aggregation algorithm (ALG-AGG) (see Algorithm 1) which is basically composed of three stages – hit evidences integration, computation of NDT model and miss evidences integration. The 2D visual example of aggregation process is shown in Figure 3.5.

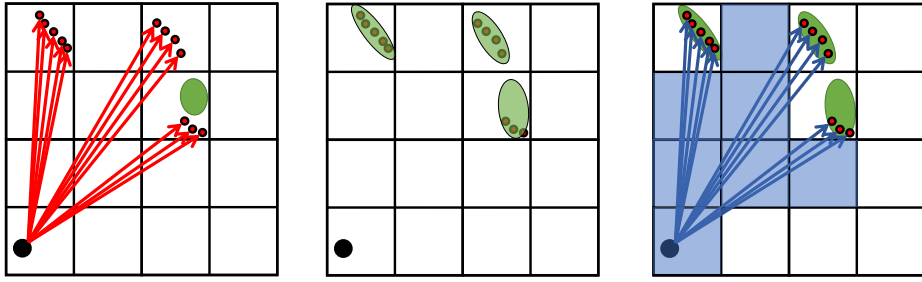


Figure 3.5: An illustration of 3 stages of aggregation process. (Left) At first, hit evidences – red dots – are integrated into voxels they belong to. (Middle) Then NDT models – green ellipses – are computed (two upper voxels) or recomputed (lower voxel) for each voxel which contains new hit evidences. (Right) At the end, miss evidences – blue squares – are integrated.

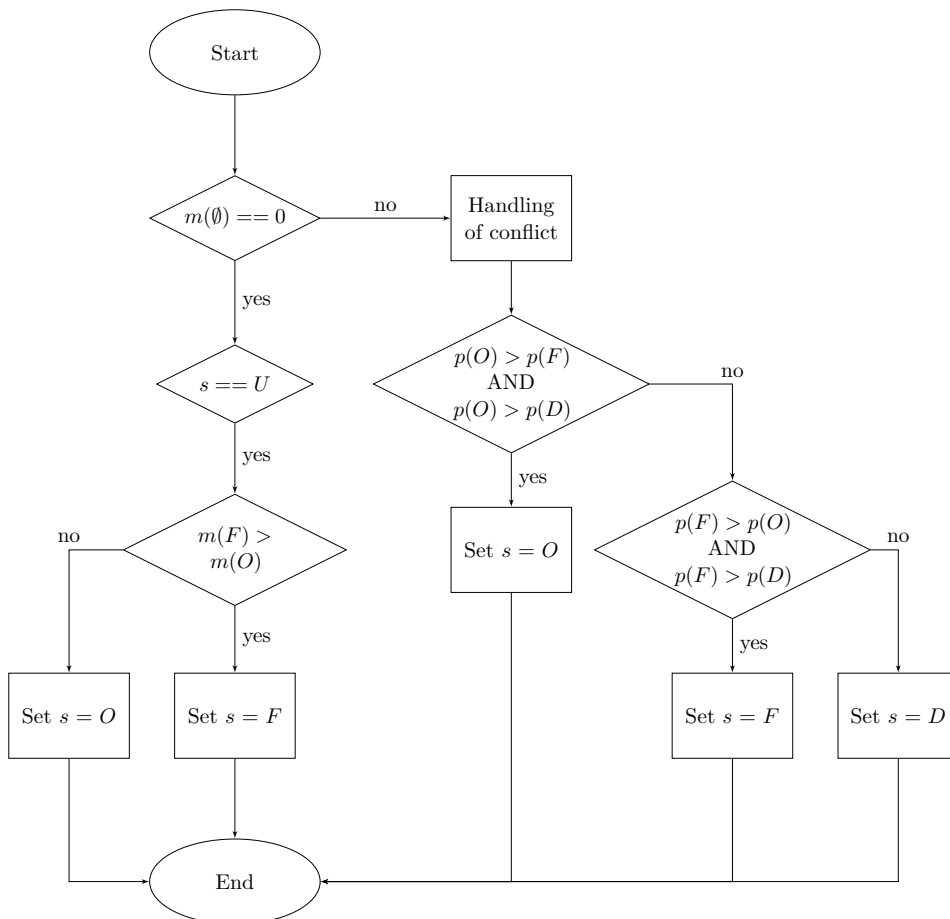


Figure 3.6: Flowchart of node's state updating process.

Algorithm 1 ALG-AGG

Input: point cloud \mathcal{P} , NDT-BM octree \mathcal{O} , source 3D positions \mathcal{S} , parameter τ_R **Output:** octree \mathcal{O}

```

1: if  $\mathcal{O}$  is null then
2:   initialize  $\mathcal{O}$  as new NDT-BM octree;
3: end if
4: for all  $x_i \in \mathcal{P}$  do ▷ Hit evidences integration
5:   Find node  $n_j \in \mathcal{O}$  within which  $x_i$  lies;
6:    $B_j \leftarrow$  conjunction of  $B_j$  with  $B_{\text{hit}}$ ;
7:    $P_j \leftarrow$  compute pignistic probabilities from  $B_j$ ;
8:    $\alpha_j \leftarrow \alpha_j + 1$ ;
9:   Update state of  $n_j$  using process from Figure 3.6;
10:   $X_j \leftarrow X_j \cup x_i$ ;
11: end for
12: for all  $n_i \in \mathcal{O}$  and  $n_i$  is leaf node do ▷ Computation of NDT
13:   Compute parameters of  $N_i$  (see 3.2);
14:    $X_i \leftarrow \emptyset$ ;
15: end for
16: for all  $x_i \in \mathcal{P}$  do ▷ Miss evidences integration
17:   Take  $x_s$  3D position of corresponding source from  $\mathcal{S}$ ;
18:   Cast ray  $r_i$  from  $x_s$  to  $x_i$ ;
19:   for all  $n_j \in \mathcal{O}$  and  $r_i$  goes through  $n_j$  do
20:      $B_{\text{tmp}} \leftarrow$  conjunction of  $B_j$  with  $B_{\text{miss}}$ ;
21:     if  $m(\emptyset)_{\text{tmp}} > 0$  then
22:       Compute relative distance  $d_R$  between  $r_i$  and  $N_j$  (see 3.2.3);
23:       if  $d_R > \tau_R$  then
24:         continue;
25:       end if
26:     end if
27:      $B_j \leftarrow B_{\text{tmp}}$ 
28:      $P_j \leftarrow$  compute pignistic probabilities from  $B_j$ ;
29:      $\beta_j \leftarrow \beta_j + 1$ ;
30:     Update state of  $n_j$  using process from Figure 3.6;
31:   end for
32: end for

```

Chapter 4

Change detection in vehicle neighborhood

The second part of our work is to develop method for on-line segmentation of new 3D measurements into known scene and a new objects. As in previous chapter, we would like to introduce overall flow of change detection process (see Figure 4.1), before the details are described.

Again, vehicle is equipped with set of LiDARs and navigation platform. A reduced version of the 3D map (described in chapter 3) for desired part of the environment is available. During the ride, LiDAR continuously collects its 3D measurements until it made one rotation, then the collection of raw points is transformed into global coordinate system. These registered 3D points, denoted as *packet*, are processed by proposed change detection algorithm (ALG-DET), which use knowledge about the environment from received 3D map to segment points in input packet to those, which are *close* enough to some known object in the map, and those which are too *far* from all objects in the map. This process is repeated during the whole ride.

4.1 3D map reduction

To decrease memory usage and processing time, we separate subset of voxels M_{occ} from input 3D map M (created by ALG-AGG), which are occupied, and which contains enough hit evidences (i.e., $\alpha \geq \alpha_{min}$). Then we create an instance of kd-tree data structure K which nodes corresponds to the mean value μ_i stored in N_i property of every voxel $v_i \in M_{occ}$. Moreover, in each node we save covariance matrix C as auxiliary information.

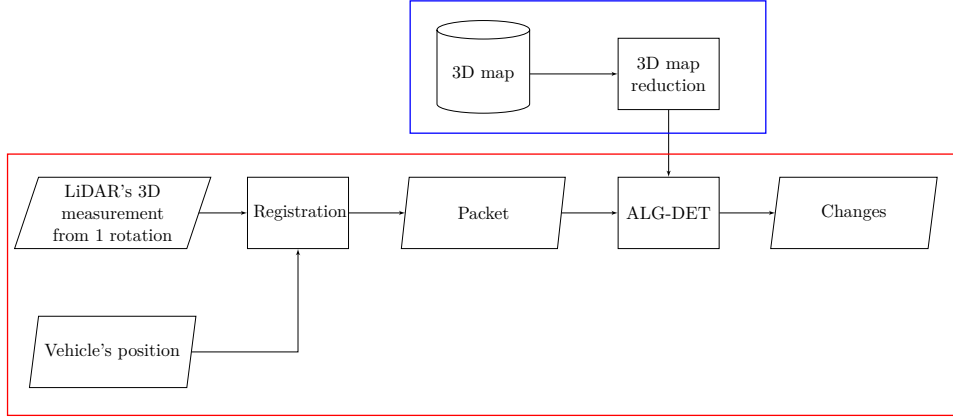


Figure 4.1: Overall workflow of change detection process. At first, 3D measurements from one LiDAR’s rotation are registered into global coordinate system (*packet*). Then change from packet’s points is detected proposed change detection algorithm (ALG-DET). This process is repeated during the whole ride. Red and blue frame denote the part of the process running on the side of the vehicle and server, respectively.

4.2 Points segmentation

Mahalanobis segmentation. Each input packet \mathcal{P} we need to divide into two subsets $\mathcal{P}_{\text{close}}$ (points which are classified as close) and \mathcal{P}_{far} (points which are classified as far), while $\mathcal{P} = \mathcal{P}_{\text{close}} \cup \mathcal{P}_{\text{far}}$, $\mathcal{P}_{\text{close}} \cap \mathcal{P}_{\text{far}} = \emptyset$.

Basic idea how to do that is to find for each point $\vec{p}_i \in \mathcal{P}$ closest Gaussian in kd-tree K and by thresholding decide if its close enough ($\vec{p}_i \in \mathcal{P}_{\text{close}}$) or not ($\vec{p}_i \in \mathcal{P}_{\text{far}}$). To determine the distance between point \vec{p}_i and 3D gaussian (normal) distribution represented by mean μ and covariance matrix C we chose Mahalanobis function [Mah36],

$$d_M(\vec{p}_i, \vec{\mu}) = \sqrt{(\vec{p}_i - \vec{\mu})^T C^{-1} (\vec{p}_i - \vec{\mu})}. \quad (4.1)$$

To obtain inverse of covariance matrix we use process from subsection 3.2.2.

Euclidean segmentation. The calculation of d_M and C^{-1} for each of the point \vec{p}_i may have negative impact on the speed of the change detection. For this reason, we decide to add filtering step which will precede to Mahalanobis segmentation and which is based on following assumption. Kd-tree K is constructed directly from voxels in map M , so we can assume that every node in K represents distribution of mass in the same position and within the same boundaries as gaussians in voxels. And because of that, if the Euclidean distance

$$d_E(\vec{p}_i, \vec{\mu}) = \sqrt{(\vec{p}_i - \vec{\mu})^T (\vec{p}_i - \vec{\mu})}. \quad (4.2)$$

between point \vec{p}_i and mean of the closest Gaussian distribution in K is greater than parameter $\tau_E^{\text{upper}} \geq \delta$, we can say that this point \vec{p}_i do not belong to any static object in map M . Also, if d_E between point \vec{p}_i and nearest

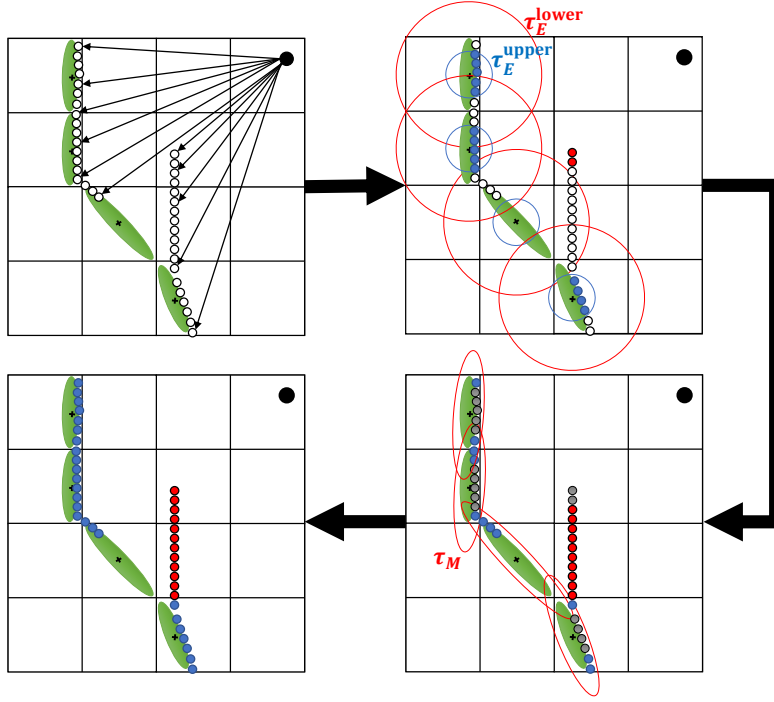


Figure 4.2: An illustration of points segmentation process. On the top left image, input packet with non-labeled points (marked as white dots) and 3D map (as green ellipses) are shown. Euclidean segmentation is shown in top right image, where points, distant more than τ_E^{upper} , are labeled as *far* (red dots) and points, distant less than τ_E^{lower} , are labeled as *close* (blue dots). Mahalanobis segmentation is illustrated in the bottom right image, where points, distant more than τ_M , are labeled as *far*, the rest as *close* (gray dots highlight already classified points). In the bottom left image, result of segmentation is shown. Each point is either classified as *close* - blue dots - or as *far* - red dots, is in bottom left image. Note, the grid is shown only for presentation purpose.

Gaussian distribution in K is lower than parameter $\tau_E^{\text{lower}} < \delta$, we can assume this point \vec{p}_i is reflection from static object in map M .

4.3 Change clustering and tracking

After registration step, packet can contains separate points or even whole stripes of incorrectly transformed raw measurements which can be falsely segmented as change. To make our change detection method more robust, we decided to do following.

Clustering. After the segmentation of packet described in section 4.2, we take points in \mathcal{P}_{far} and cluster them using algorithm DBSCAN [EKSX96] (which requires two parameters ϵ and p_{min}). As its output, we get label for each point, to which cluster $s_i \in S$ it belongs, or if the point is noise. For

each cluster $s_i \in S$, we compute normal distribution parameters, mean μ_i and covariance matrix C_i (see section 3.2), which will serve as cluster's descriptor $\mathcal{N}_i(\mu_i, C_i)$.

Tracking. By clustering in previous paragraph, outliers are eliminated. To detect and discard groups of points which occur in packets irregularly, we will check if object represented by cluster $s_i \in S$ is appeared in previous packets or not. To be able to track recurrent clusters, we will keep in memory all clusters S^{prev} obtained from the previously processed packet and. Then for every $s_i \in S$ we find the least distant cluster $s_{\min}^{\text{prev}} \in S^{\text{prev}}$. If their distance is lower than the threshold τ_B , we consider that both clusters belong to the same object and points in s_i can be reported as change. Otherwise, we cannot be sure if s_i represents new object, which is observed by LiDAR for the first time, or if these points are error.

The similarity between two clusters s_A and s_B represented by their descriptors $\mathcal{N}_A(\mu_A, C_A)$ and $\mathcal{N}_B(\mu_B, C_B)$ is based on Bhattacharyya distance [Bha46] between two multivariate normal distributions,

$$d_B(s_A, s_B) = \frac{1}{8}(\mu_A - \mu_B)^T C^{-1}(\mu_A - \mu_B) + \frac{1}{2} \ln \left(\frac{\det C}{\sqrt{\det C_A \det C_B}} \right), \quad (4.3)$$

where $C = \frac{C_A + C_B}{2}$.

4.4 Implementation details

The classification process to detect changes in car's surrounding described in previous sections is illustrated in Figure 4.2 and is summarized into algorithm ALG-DET (Algorithm 2).

Algorithm 2 ALG-DET

Input: packet \mathcal{P} , 3D map \mathcal{M} , clusters S^{prev} , parameters $k, \tau_B, \tau_E^{\text{upper}}, \tau_E^{\text{lower}}, \tau_M$ and α_{\min} **Output:** set of 3D points $\mathcal{P}_{\text{close}}$ and \mathcal{P}_{far}

- 1: Create empty kd-tree K ▷ 3D map preparation
- 2: **for all** $m_i \in \mathcal{M}$ **do**
- 3: **if** $s_i = O$ **and** $\alpha_i \geq \alpha_{\min}$ **then**
- 4: Create element e with position μ and with auxiliary information C ;
- 5: Insert e into K ;
- 6: **end if**
- 7: **end for**
- 8: Initialize $\mathcal{P}_{\text{close}} = \emptyset$ and $\mathcal{P}_{\text{far}} = \emptyset$; ▷ Points segmentation
- 9: **for all** $\vec{p}_i \in \mathcal{P}$ **do**
- 10: $T \leftarrow$ find k -nearest nodes to \vec{p}_i from K ;
- 11: $d_{\min} \leftarrow$ nearest node to \vec{p}_i from T ;
- 12: **if** $d_{\min} > \tau_E^{\text{upper}}$ **then**
- 13: $\mathcal{P}_{\text{far}} \leftarrow \mathcal{P}_{\text{far}} \cup \vec{p}_i$;
- 14: **continue**;
- 15: **end if**
- 16: **if** $d_{\min} < \tau_E^{\text{lower}}$ **then**
- 17: $\mathcal{P}_{\text{close}} \leftarrow \mathcal{P}_{\text{close}} \cup \vec{p}_i$;
- 18: **continue**;
- 19: **end if**
- 20: **for all** $t_j \in T$ **do**
- 21: Compute Mahalanobis distance d_M between \vec{p}_i and N_j ;
- 22: **if** $d_M < \tau_M$ **then**
- 23: $\mathcal{P}_{\text{close}} \leftarrow \mathcal{P}_{\text{close}} \cup \vec{p}_i$;
- 24: **continue**;
- 25: **end if**
- 26: **end for**
- 27: $\mathcal{P}_{\text{far}} \leftarrow \mathcal{P}_{\text{far}} \cup \vec{p}_i$;
- 28: **end for**
- 29: $S \leftarrow$ DBSCAN(\mathcal{P}_{far}); ▷ Change clustering and tracking
- 30: **for all** $s_i \in S$ **do**
- 31: $d \leftarrow \min_{\forall s_k \in S^{\text{prev}}} d_B(s_i, s_k)$;
- 32: **if** $d < \tau_B$ **then**
- 33: report s_i ;
- 34: **end if**
- 35: **end for**

Chapter 5

Experiments

In our work we proposed two different algorithms. The aim of the first one (ALG-AGG) was to create 3D map of car's surrounding using information from LiDAR sensor (chapter 3). And the purpose of the second one (ALG-DET) was to detect changes in the car's environment (chapter 4) using the 3D map (output of the ALG-AGG).

To be able to evaluate correctness of proposed algorithms, two different datasets were selected – raw data from *the KITTI Vision Benchmark Suite* (KITTI) [GLSU13] and *The University of Michigan North Campus Long-Term Vision and LIDAR Dataset* (NCLT) [CBUE15].

5.1 Description of used datasets

Both used datasets have several features in common. Their data were obtained by moving vehicle in real-world urban environment. The vehicle was equipped among others with LiDAR and navigation sensors (providing vehicle's position during the time) and are freely available online.

Each of the dataset consists of several *recordings*. Every such recording represent one ride of vehicle and contains set of LiDAR's 3D measurements called *point cloud*.

5.1.1 KITTI

From KITTI, we selected recording 2011_09_26_drive_0017 (KITTI-17) and 2011_09_26_drive_0018 (KITTI-18). Both recordings capture the same

Recording	Map part	Shortcut
2012-02-04	P01	NCLT-02-P01
	P02	NCLT-02-P02
2012-04-29	P01	NCLT-04-P01
	P02	NCLT-04-P02
2012-06-15	P01	NCLT-06-P01
	P02	NCLT-06-P02
2012-09-28	P01	NCLT-09-P01
	P02	NCLT-09-P02

Table 5.1: Selected NCLT recordings

crossroads. In the first one, car is waiting at the traffic lights. In the second recording, after a while, the green light is turned on and the car turns left.

■ 5.1.2 NCLT

The main reason why we decided to use NCLT dataset is the fact, that its recordings are captured in the same area but in different time (in the range of 15 months). Because of that, some changes are present in the area, and it is possible to test the behavior of the data aggregation algorithm (see chapter 3) when data from multiple rides are available (in contrast with KITTI) and see how the changes in urban environment are reflected in the aggregated map.

For our experiments we chose four recordings (see Table 5.1) and we selected two small parts from the whole map, which we found suitable for testing purpose. First part, denoted as P01, is a straight path between two buildings, whereas one of them is under construction, which cause arising of new static objects (e.g., fence). Second part, P02, is a straight road with pavements on its both sides, which yields occurrences of dynamic objects (e.g., cars, pedestrians).

■ 5.2 Parameters setup

In both proposed methods, several parameters were defined. They are listed in Table 5.2 with assigned values, which were used during all experiments presented in this chapter, unless otherwise stated.

■ 5.3 Proposed methods evaluation

To evaluate correctness of ALG-AGG and ALG-DET we prepared several test scenarios which should simulate some of the situations which could occur when our methods were used in real autonomous car.

Parameter	Value	Description
ρ_L	0.25	lower and upper threshold defining interval in witch ρ is taken as similar for both evidence type
ρ_U	0.50	
λ_O	0.60	parameters of Gaussian function for $m(O)_{\text{hit}}$
μ_O	1	
σ_O	0.20	
λ_D^{hit}	0.40	parameters of Gaussian function for $m(D)_{\text{hit}}$
μ_D	0.38	
σ_D^{hit}	0.10	
$\lambda_{OD}^{\text{hit}}$	0.40	parameters of Gaussian function for $m(\{O, D\})_{\text{hit}}$
μ_{OD}	0.38	
σ_{OD}^{hit}	0.10	
λ_F	0.60	parameters of Gaussian function for $m(F)_{\text{miss}}$
μ_F	0	
σ_F	0.20	
λ_D^{miss}	0.25	parameters of Gaussian function for $m(D)_{\text{miss}}$
μ_D	0.38	
σ_D^{miss}	0.10	
$\lambda_{OD}^{\text{miss}}$	0.10	parameters of Gaussian function for $m(\{O, D\})_{\text{miss}}$
μ_{OD}	0.38	
$\sigma_{OD}^{\text{miss}}$	0.10	
ϵ	0.75	parameters of DBSCAN algorithm
p_{min}	10	
α_{min}	20	minimal number of hit evidence in voxel to be used as map element in change detection
δ	0.50	resolution of NDT-BM's grid
τ_B	0.35	threshold used in Mahalanobis segmentation
τ_E^{lower}	0.5	lower threshold used in Mahalanobis segmentation upper threshold threshold used in Mahalanobis segmentation
τ_E^{upper}	0.08	
τ_M	8	threshold used in Mahalanobis segmentation
τ_R	0.5	threshold used for relative distance between ray and mass
ω	1000000	lowest possible resolution for inverse covariance matrix
K	3	number of nearest neighbors, used in ALG-DET

Table 5.2: List of all used parameters and their values used in proposed methods.

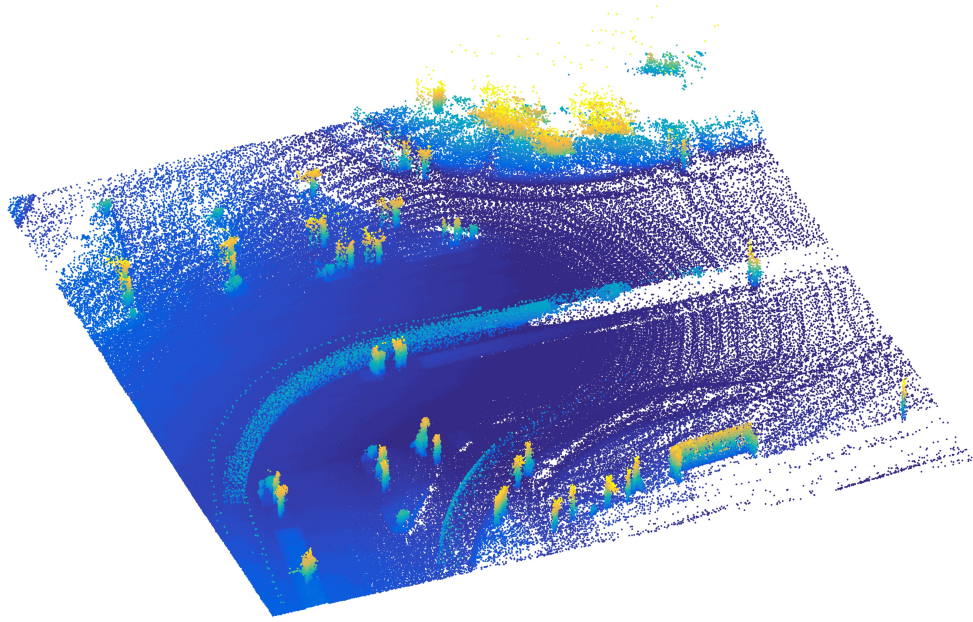


Figure 5.1: Visualization of the input point cloud (10% of points) of the recording KITTI-18. There are two obvious track caused by moving vehicles.

■ 5.3.1 Test scenario 1 (TS1)

Description. Purpose of TS1 to evaluate the correctness of the creation of the initial 3D map using ALG-AGG and data from single ride through the urban environment as its input. The condition for scenario success is that moving objects in appropriate recording will not be included in the output 3D map, but all static object will be.

Evaluation. To run TS1, we selected recording, KITTI-18 (see Figure 5.1). By comparing point cloud and aggregated map for KITTI-18 (see Figure 5.2 we see, that the ground forms compact surface. In the middle of the map there is small trace leaved by moving cars. In this part of the map, our car was waiting on traffic lights and several other cars were standing behind it. When our car started move forward, it was followed by these other cars, which bodies prevented LiDAR to shoot its ray through this part. Thus, there was much more hit evidences then the miss evidences. However, one issue arise from this scenario. KITTI-18 contains a lot of vertical objects with width smaller than voxel's resolution (street lamps, traffic lights etc.). And in several cases their bottom parts were classified as *dynamic* (see Figure 5.3).

■ 5.3.2 Test scenario 2 (TS2)

Description. Scenario TS2 should test how the 3D map is evolving over time when appropriate recordings are iteratively used to update it (via ALG-AGG).

Evaluation. For TS2, we chose recordings NCLT-02-P01, NCLT-04-P01 and NCLT-06-P01, because of changes which happen over time (see Figure 5.4).

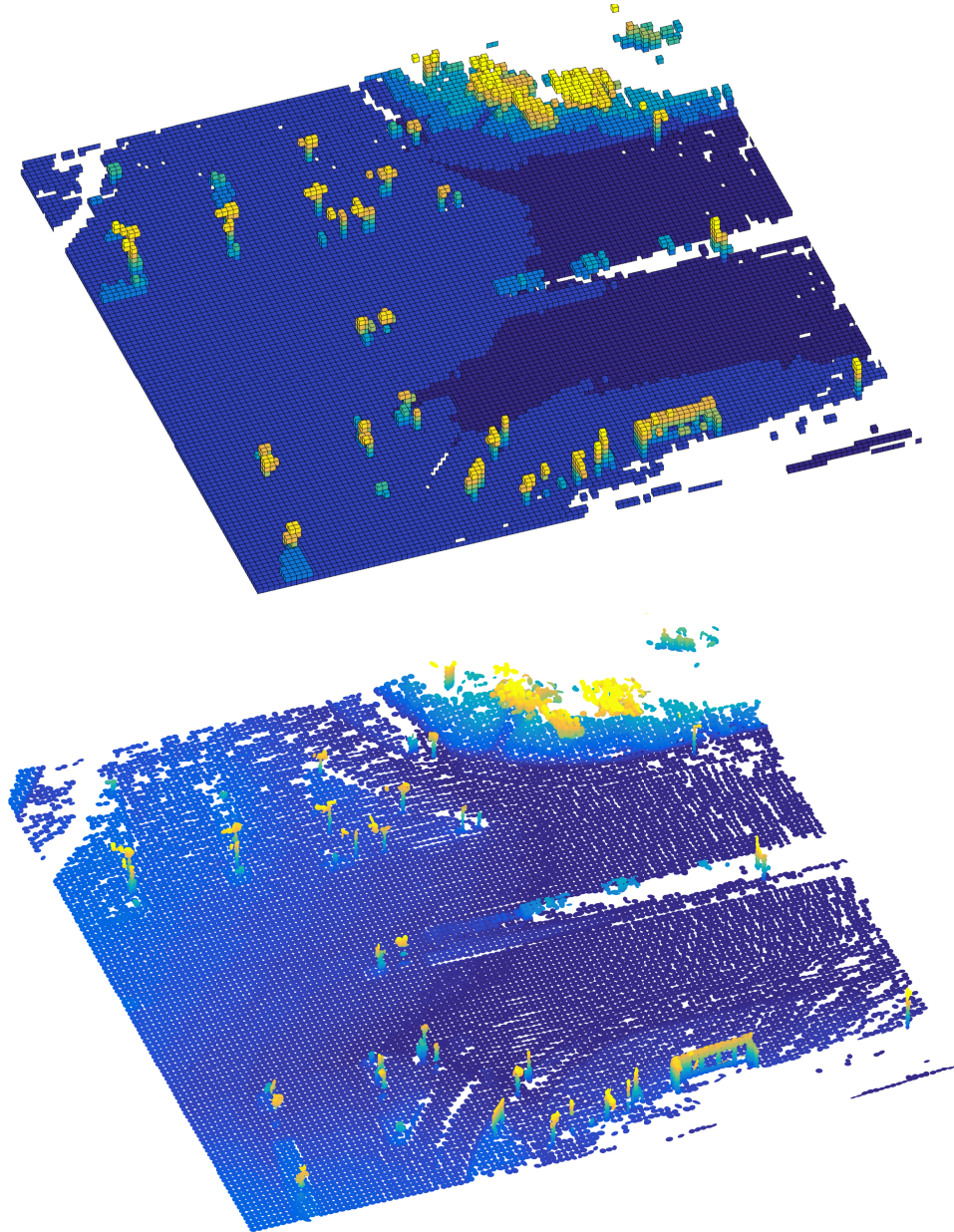


Figure 5.2: NDT-BM for recording KITTI-18. Occupied voxels are shown in the top picture and their 3D Gaussian's are visualized on the bottom picture

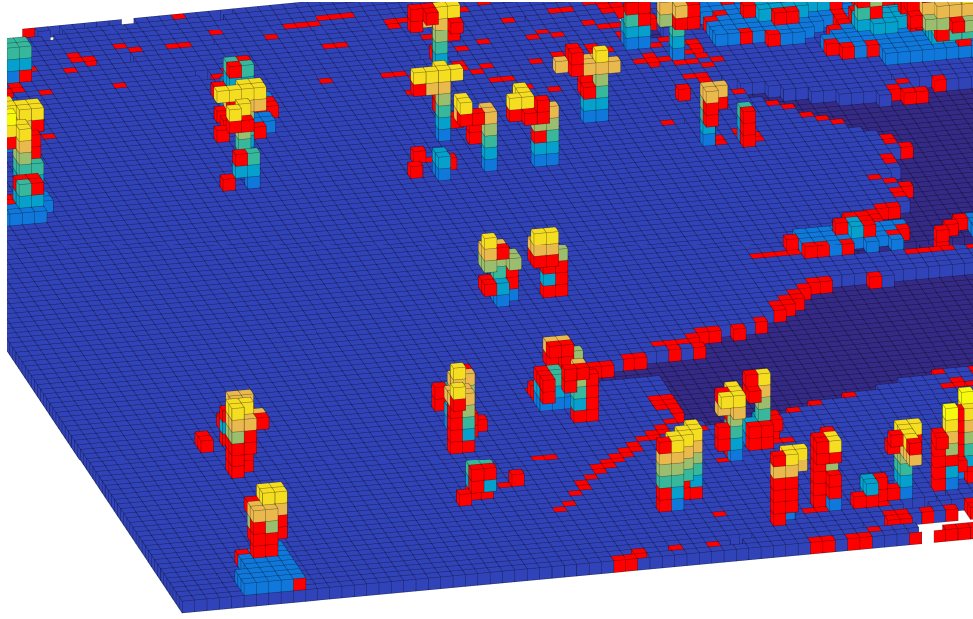


Figure 5.3: Detail of the 3D map from Figure 5.2. Red cubes corresponds to dynamic voxels, the others to occupied one.

As we can see, maps for all three recordings mostly correspond to their inputs. Notice, that the new roof from the recording NCLT-06-P01 were integrated as occupied just after one ride, but new walls occurring in recordings NCLT-04-P01 and NCLT-06-P01 are still labeled as dynamic. Until recording NCLT-06-P01, LiDAR's rays went through this part without hitting any object (i.e., did not return reflecting point), thereby there were not any evidence. Thus, the roof were integrated faster than the walls.

There is another interesting point. In the right part of the NCLT-02-P01, there are some trees and shrubs. Trees were more or less classified as occupied, but shrubs were mostly classified as dynamic (see Figure 5.6). This inconsistency can be caused by fact, that shrubs have smaller, thinner branches than trees. So they can be more easily moved by wind to change their position which leads to dissimilar evidences. With increasing number of integrated recording, relevant voxels become occupied.

■ 5.3.3 Test scenario 3 (TS3)

Description. The purpose of the TS3 is to verify if ALG-DET correctly detects new *static* objects, which appeared in the known scene represented by NDT-BM.

Evaluation. As the input 3D map, we used the one created from NCLT-02-P01, and as input recording, we used NCLT-04-P01. By comparing their point clouds (see Figure 5.4), there are several new static obstacles in NCLT-04-P01.

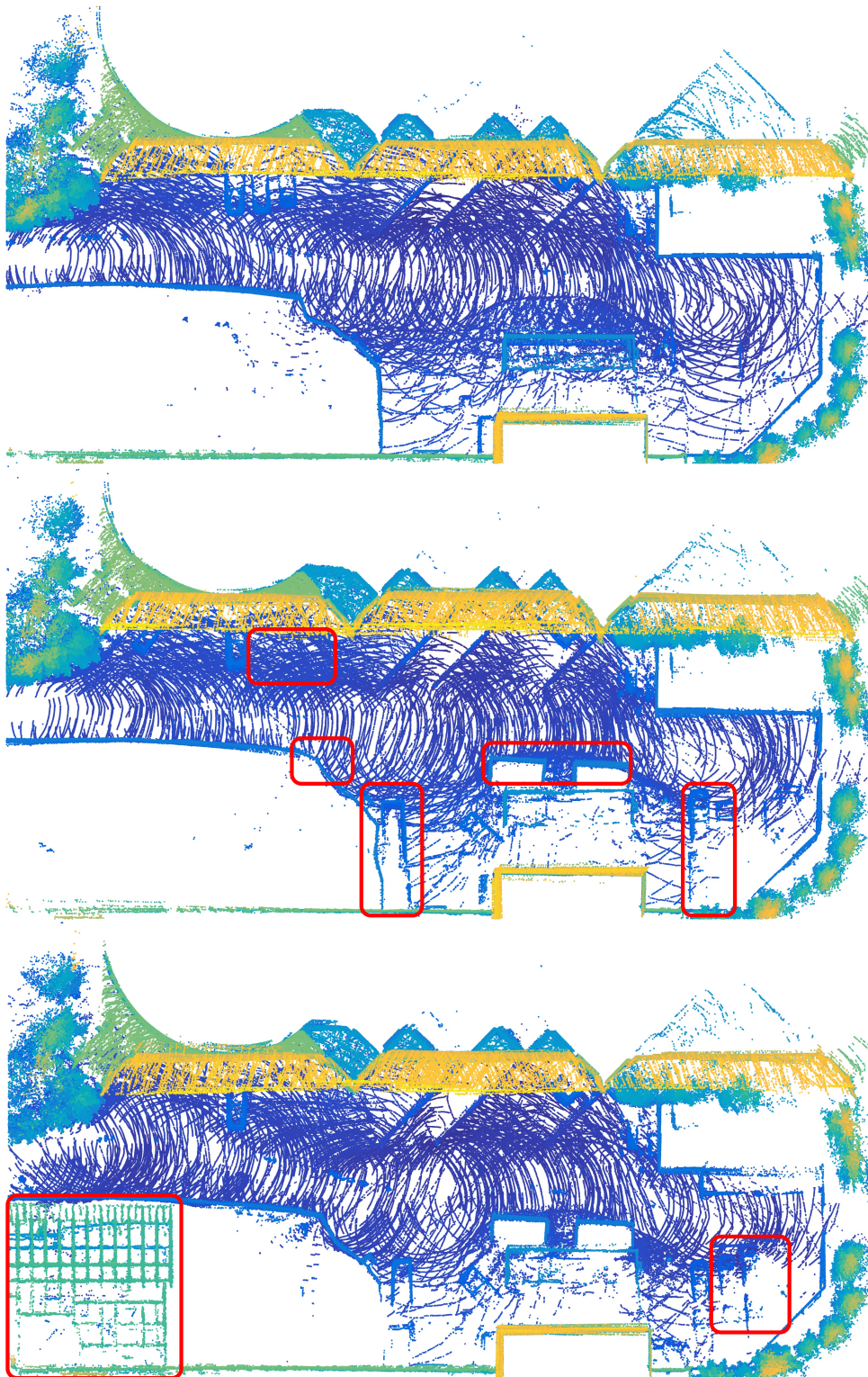


Figure 5.4: Visualization of input point clouds for recordings (top) NCLT-02-P01, (middle) NCLT-04-P01 and (bottom) NCLT-06-P01. In the middle and bottom picture, static changes (compared to previous recording) are emphasized by red frames.

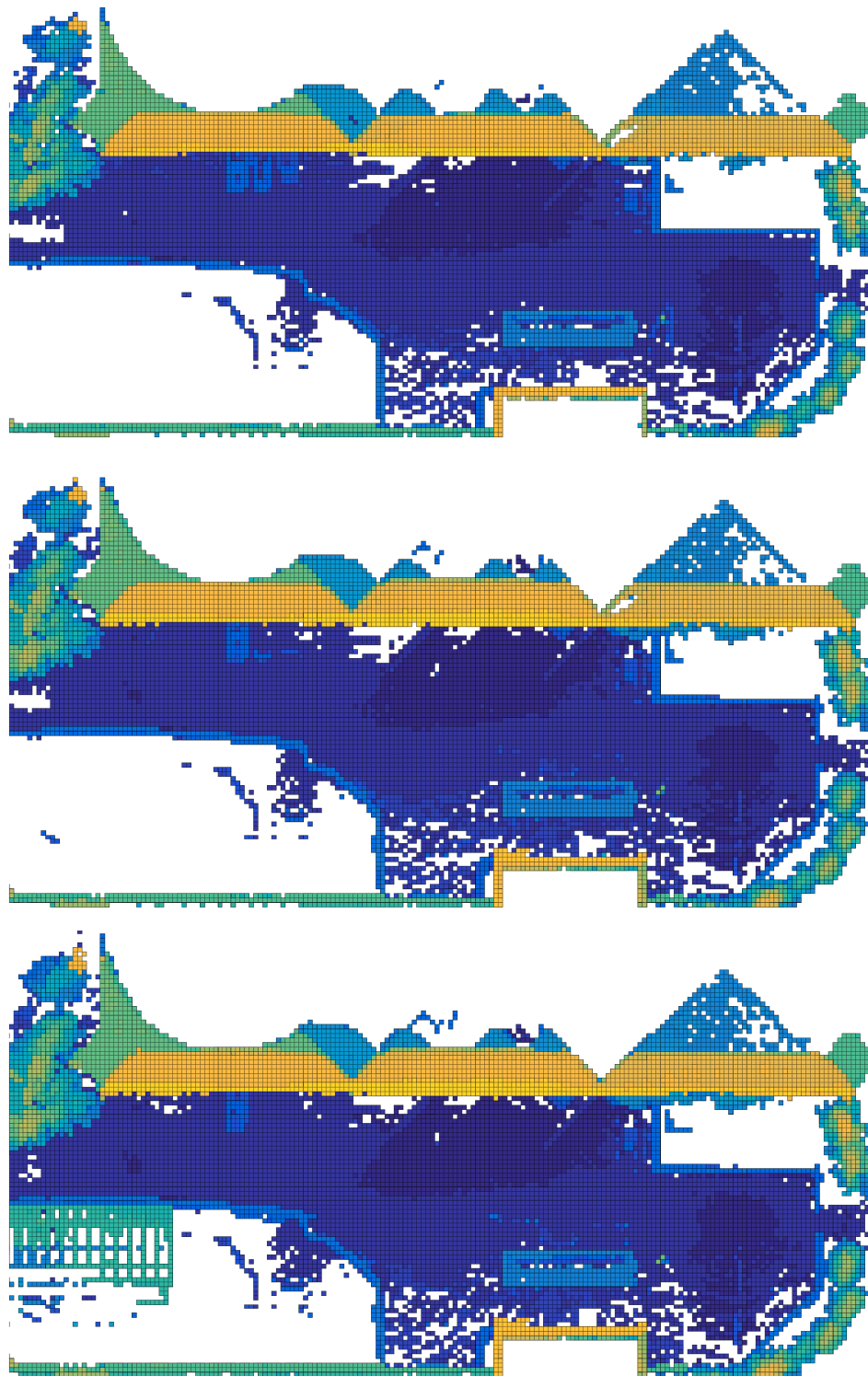


Figure 5.5: Evolution of NDT-BM created from recordings (top) NCLT-02-P01, (middle) NCLT-04-P01 and (bottom) NCLT-06-P01. The pictures shows occupied voxels only.

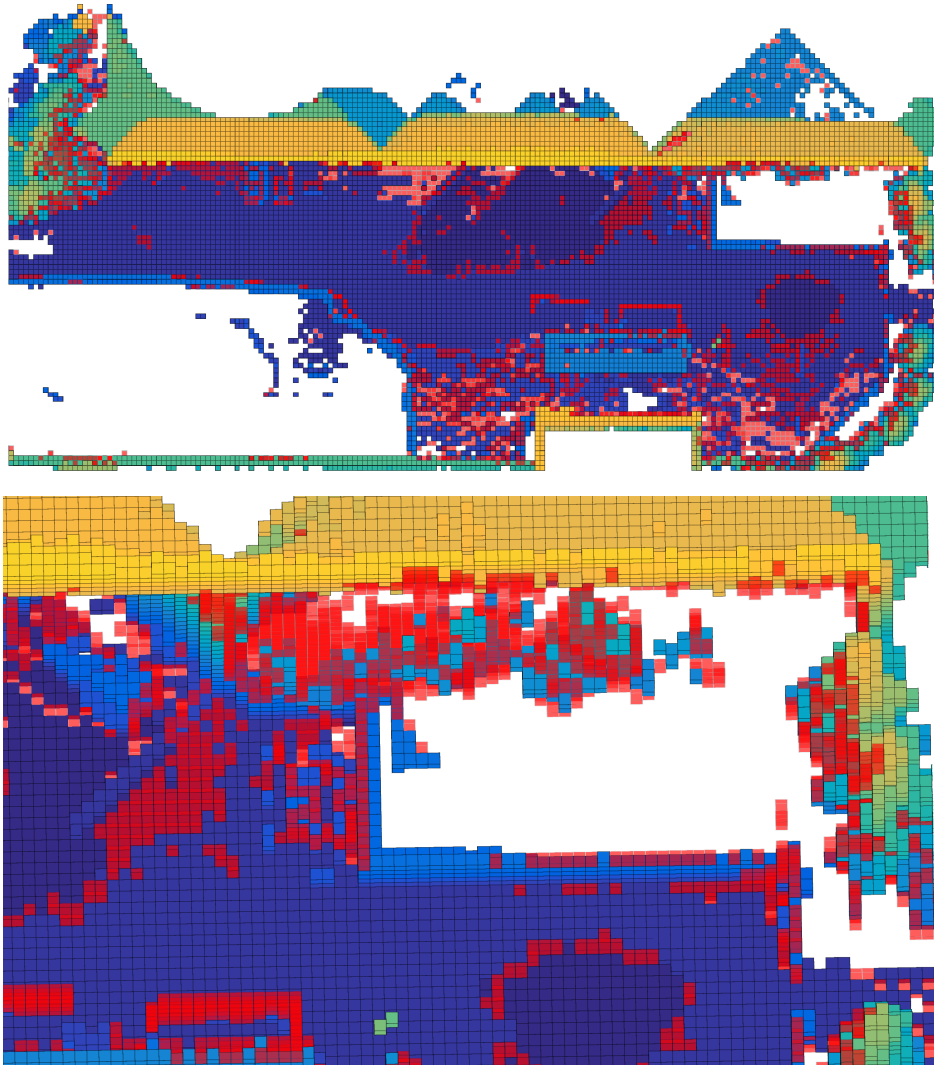


Figure 5.6: NDT-BM generated from multiple recordings (NCLT-02-P01 and NCLT-04-P01). Both pictures shows dynamic (semi-transparent red cubes) and occupied (colorful) voxels. On bottom picture is detail of the map from top picture.

After running ALG-DET and analyzing its results (see Figure 5.7), it is obvious that all of these object were successfully detected. One thing is worth mentioning. As can be seen, in the top-right part of the map changes were detected. This is related to 5.3.2, where we mentioned the problem with aggregation of shrubs. In this case, the change detection itself is not incorrect.

■ 5.3.4 Test scenario 4 (TS4)

Description. Scenario TS4 should verify if ALG-DET correctly detects moving objects, which appeared in the known scene represented by NDT-BM.

Evaluation. For evaluation of TS4, KITTI-18 and NCLT-04-P02 were used as source to create 3D map, KITTI-17 and NCLT-09-P02 were used as source of new measurements, respectively.

As it evident from KITTI-17's point cloud (see Figure 5.8), there are several moving vehicles and one moving track. According to results of ALG-DET (see Figure 5.9 and Figure 5.10), all of these objects where successfully detected throughout the ride.

Recording NCLT-09-P02 contains moving cyclist (see Figure 5.11) which was successfully detected, clustered and tracked (see Figure 5.10).

■ 5.4 Comparing ALG-AGG with OctoMap framework

For comparison, we created 3D occupancy grid using original OctoMap (OM) framework for recording KITTI-18 and grid's resolution 0.5 m and 0.25 m.

If we compare output of our approach (see Figure 5.2) and output of OM (see Figure 5.14), there are noticeable differences. In our map, the road surface is almost completely solid and there is no trace caused by moving cars (except the voxels of cars waiting on traffic lights, discussed in subsection 5.3.1), but in both variant of OM, there are significant gaps and traces. Generating OM with lower resolution then 0.25 m could leads to better result, but the smaller the resolution is the higher is the memory consumption and processing time.

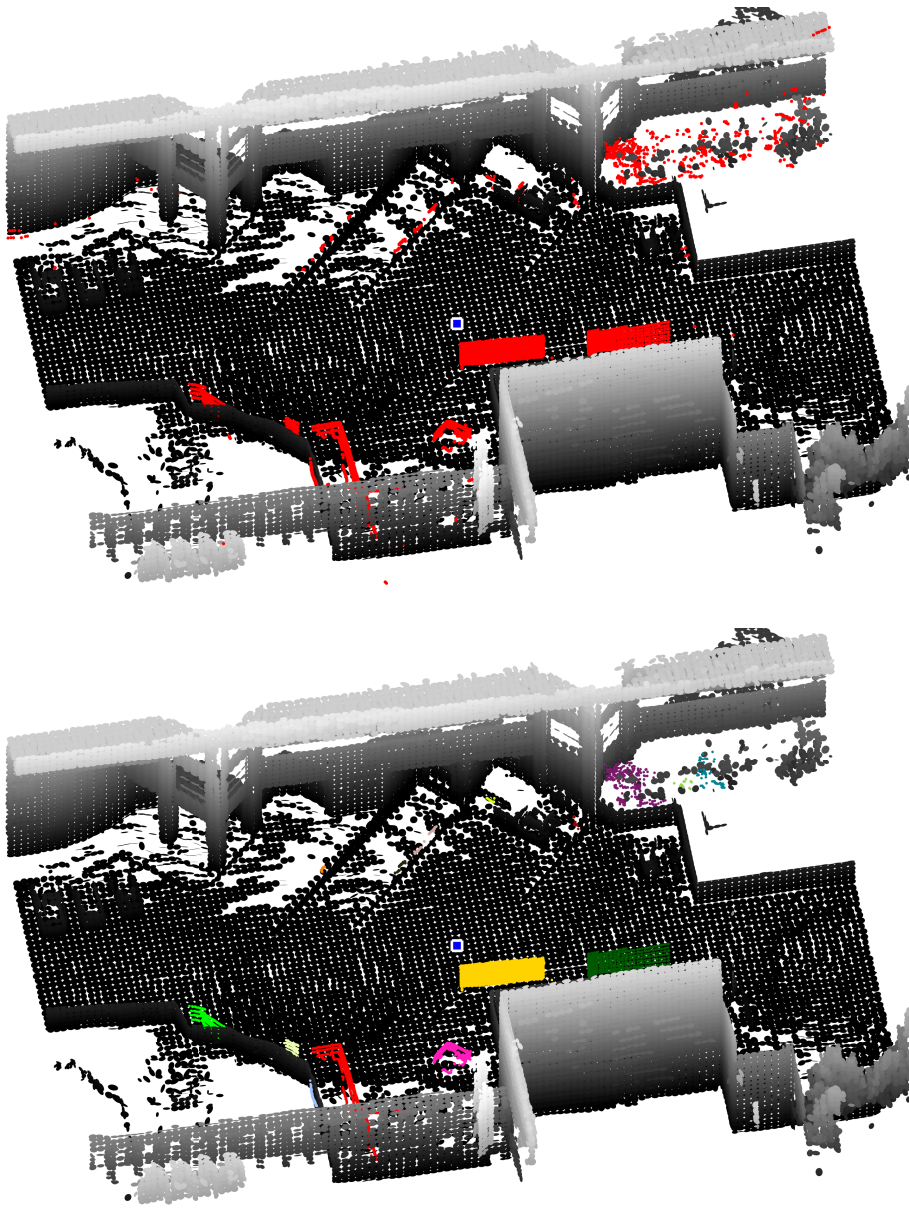


Figure 5.7: Change detection of new static objects. On top picture, 3D map aggregated from NCLT-02-P01 recording is visualized (as gray scaled ellipsoids) together with detected changes from recording NCLT-04-P01 (as red dots). The bottom picture captures same situation, but after clustering and tracking process. Changes belonging to same cluster (i.e., new object) have same color. On the both pictures, the current position of LiDAR sensor is represented by blue square with white border.

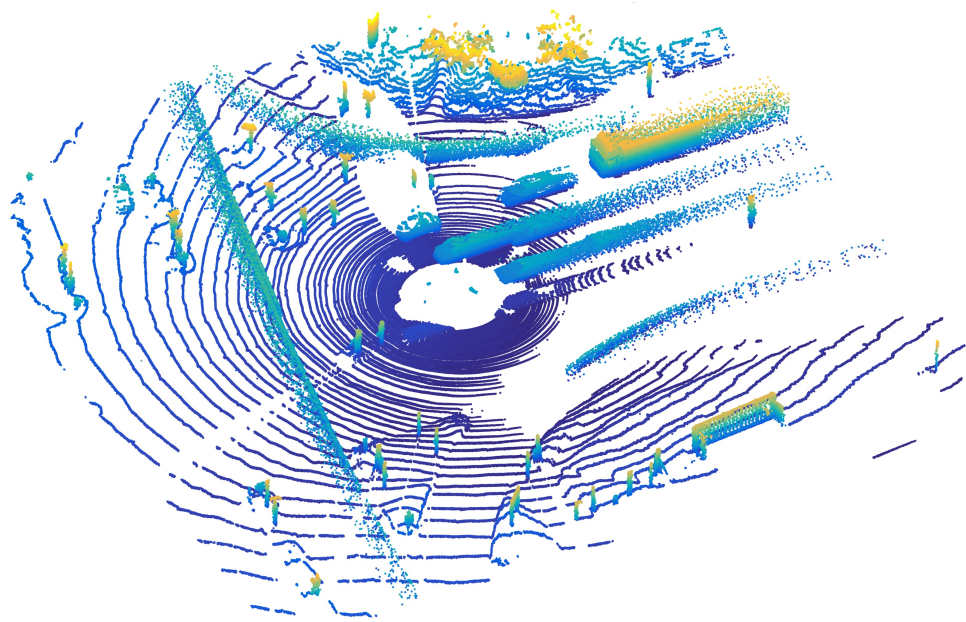


Figure 5.8: Visualization of the input point cloud (10% of points) of the recording KITTI-17. There are several tracks caused by moving cars and one caused by truck (top-right part of image).

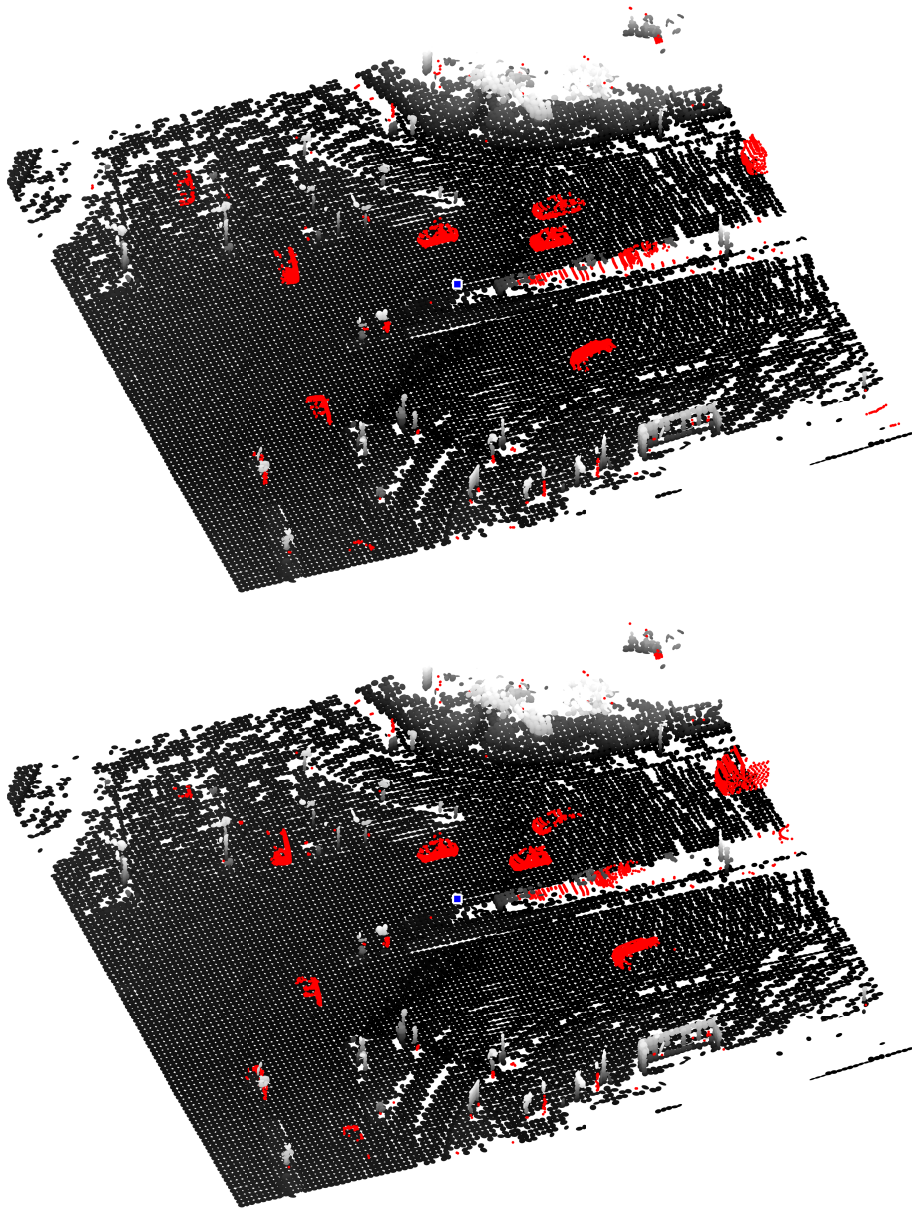


Figure 5.9: Moving objects segmentation during recording KITTI-17 using 3D map generated from recording KITTI-18. Top image correspond to recording's frame no. 3 and the bottom image to frame no. 6. Changes are visualized as red dots and the current position of LiDAR sensor as blue square with white border.

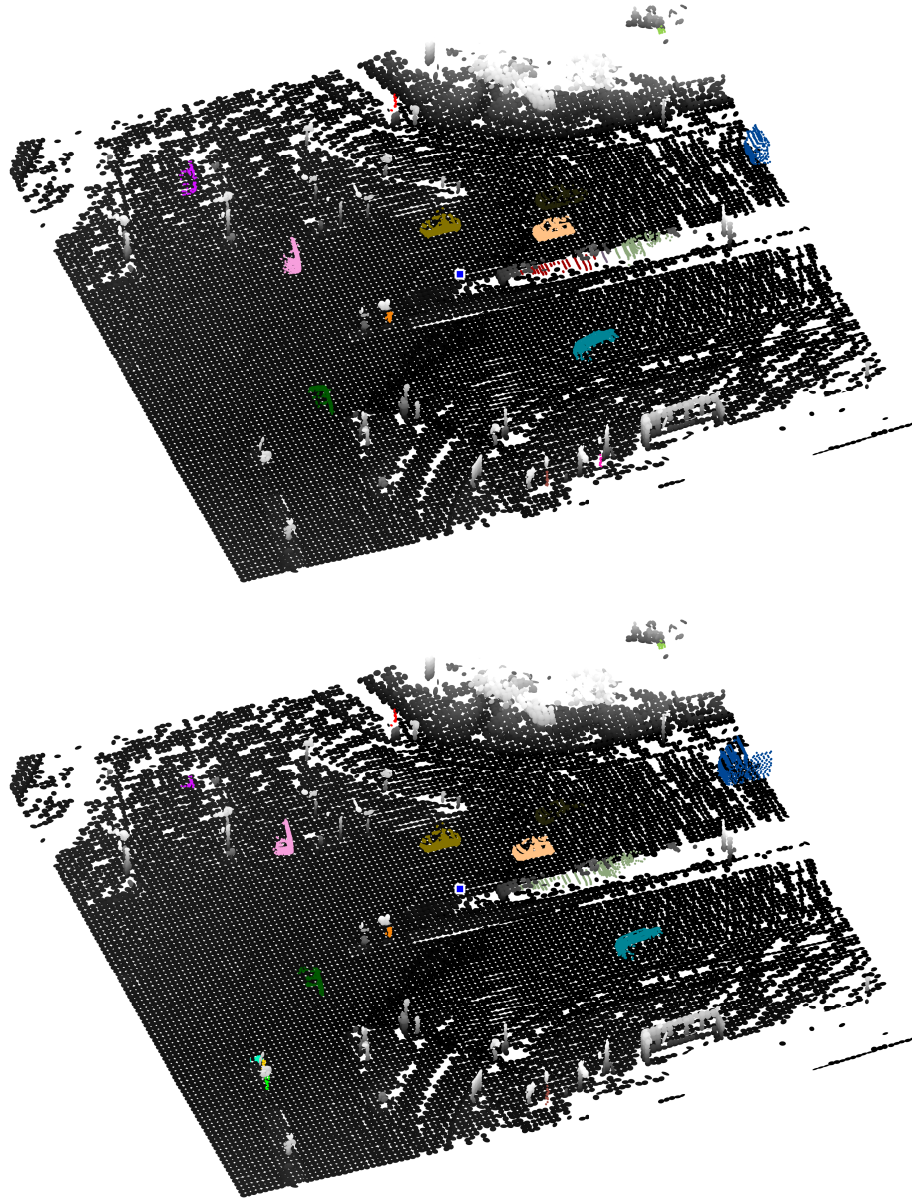


Figure 5.10: Moving objects clustering and tracking during recording KITTI-17 using 3D map generated from recording KITTI-18. Top and bottom picture captures same situation as in top and bottom picture in Figure 5.9, respectively, but after clustering and tracking process. Changes belonging to same cluster (i.e., moving object) have same color. On the both pictures, the current position of LiDAR sensor is represented by blue square with white border.

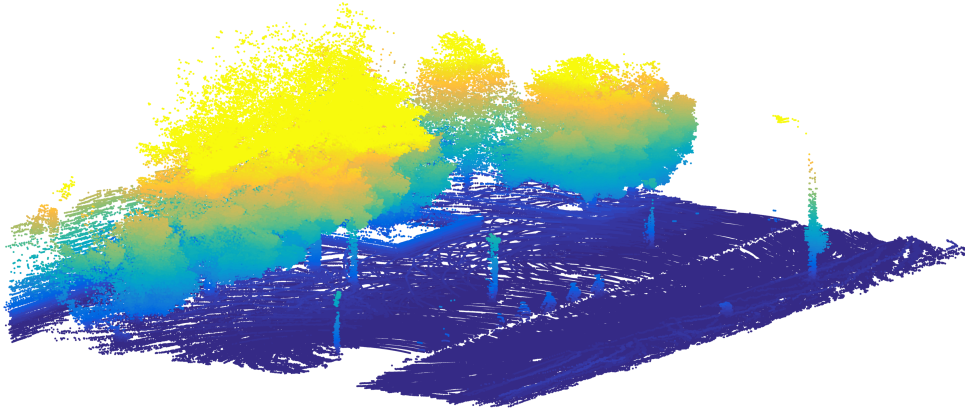


Figure 5.11: Visualization of the input point cloud (10% of points) of the recording NCLT-09-P02. There are several tracks caused by moving cars and one caused by truck (top-right part of image).

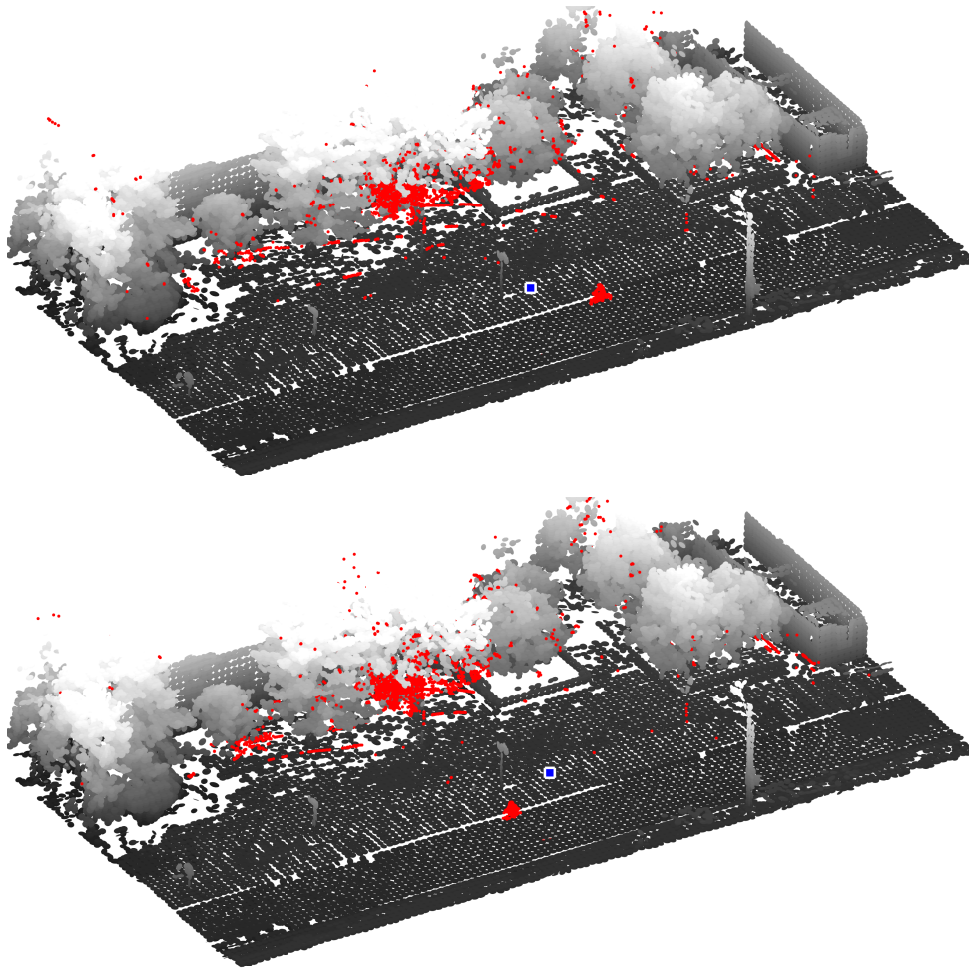


Figure 5.12: Segmentation of cyclist during recording NCLT-09-P02 using 3D map generated from recording NCLT-04-P02. Changes are visualized as red dots and the current position of LiDAR sensor as blue square with white border.

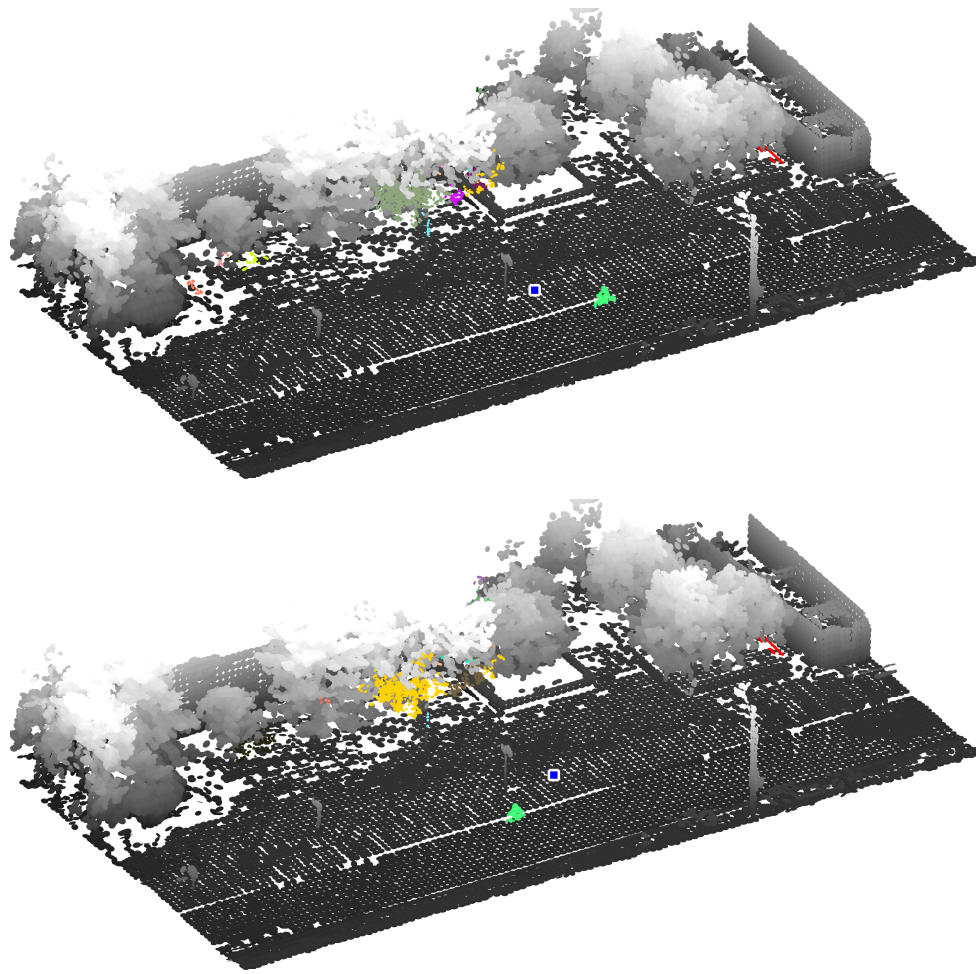


Figure 5.13: Clustering and tracking of cyclist during recording NCLT-09-P02 using 3D map generated from recording NCLT-04-P02. Top and bottom picture captures same situation as in top and bottom picture in Figure 5.12, respectively, but after clustering and tracking process.

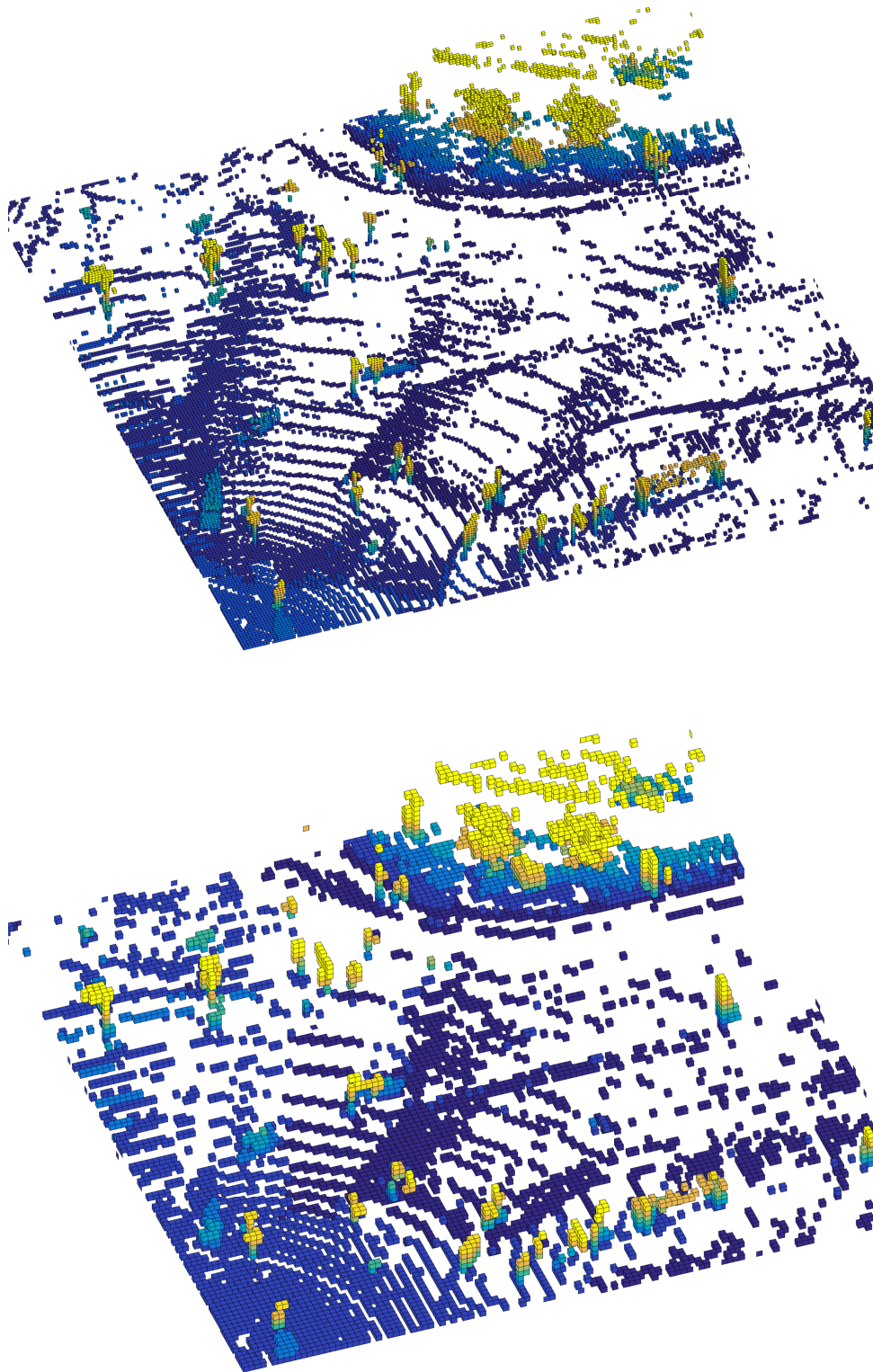


Figure 5.14: Occupancy map created by OctoMap from KITTI-18. Top picture shows result for grid with resolution 0.25 m and the bottom one shows result for grid with resolution 0.5 m.

Chapter 6

Conclusions

In this chapter, we summarize contributions and results of this thesis and discuss known limitations and future directions of our work.

6.1 Summary

At the beginning of this thesis, we described and discussed principles of related works. From this newly acquired knowledge we proposed new framework consisting of two parts, 1) how to represent and obtain 3D map of urban environment and 2) how to use this representation to detect changes in vehicle neighborhood.

In the first part, we designed new data structure for 3D map and algorithm for obtaining the map from 3D measurements. The form of our 3D map – NDT-BM – were basically derived from NDT-OM [SASL13], where occupancy mapping model were replaced by *Transferable belief model* [Sme94]. Using TBM required to design appropriate way how to generate basic belief masses for evidences and how to handle conflict, which can arise during aggregation.

In the second part, we formulated new algorithm, which aim is segmentation of new incoming 3D measurements into known scene and new objects (changes). Segmentation process is based on computation of Euclidean and Mahalanobis distances between elements of NDT-BM and new points. We also proposed procedure to overcome problem with noise in LiDAR's measurement from real-world datasets.

At the end, we proposed a series of experiments to test performance and verify correctness of our methods. The tests have shown that our digital

representation of 3D environment overcomes OctoMap and confirmed that using NDT in grid mapping tasks is useful. We verified that using our framework, it is possible to detect dynamic changes of various speed and size and static changes as well.

6.2 Limitations and open problems

From provided experiments it can be concluded, that presented methods offer decent results. Still, there are several limitations, which should not be ignored.

Results of both proposed methods are highly dependent on values of input parameters. It would be appropriate to run experiments on annotated dataset to be able to objectively compare results for different configurations. Thus, we would like to create own annotated dataset with precisely registered 3D measurements.

Proposed algorithm ALG-AGG has unsatisfying results when some vertical object with width lower than voxel's edge length needs to be mapped (e.g., street lamps in TS1). Thus, we would like to focus on modeling of this situation and refine the calculation of relative distance between mass and ray.

As was mentioned in subsection 5.3.2, there is a problem with integration of vegetation into the map. This is caused by fact, that we focused on modeling of solid objects. However, some of the LiDAR sensors are capable to return position of the strongest reflection and the latest, too, which could help to detect changeable objects like leaf, grass etc.

6.3 Future work

There are several ideas, which we would like to dedicate in the future, to enhance output of our framework.

In TS2, we show, how our NDT-BM with parameters from Table 5.2 adapts on static changes in environment. It requires future research to determine how fast should be static changes recorded into the map, but one possible way could be *Memory decay* concept [Sme07], where credibility of basic belief masses decaying over time.

To handle issue with noise in LiDAR's measurements (especially in NCLT dataset) we used heuristic approach. Therefore, it would be better to analyze this problem deeply and formulate more general solution.

NDT-BM is consists of high number of Gaussian distributions which in most cases represents part of some objects. Therefore it would be interesting to cluster distributions (e.g., by P-Linkage alg. [LYT⁺16]) which correspond to same object and then compress amount of the information, by merging each of the cluster into one representative distribution. In case of success,

this would lead to decreasing of memory requirements.



Bibliography

- [Bha46] A. Bhattacharyya, *On a measure of divergence between two multinomial populations*, Sankhyā: The Indian Journal of Statistics (1933-1960) **7** (1946), no. 4, 401–406.
- [BS03] Peter Biber and Wolfgang Straßer, *The normal distributions transform: A new approach to laser scan matching*, Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, vol. 3, IEEE, 2003, pp. 2743–2748.
- [CBUE15] Nicholas Carlevaris-Bianco, Arash K. Ushani, and Ryan M. Eustice, *University of Michigan North Campus long-term vision and lidar dataset*, International Journal of Robotics Research **35** (2015), no. 9, 1023–1035.
- [EK SX96] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu, *A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise*, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, KDD'96, AAAI Press, 1996, pp. 226–231.
- [Elf89] A. Elfes, *Using occupancy grids for mobile robot perception and navigation*, Computer **22** (1989), no. 6, 46–57.
- [GLSU13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun, *Vision meets robotics: The kitti dataset*, International Journal of Robotics Research (IJRR) (2013).
- [HWB⁺13] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard, *OctoMap: An efficient probabilistic 3D mapping framework based on octrees*, Autonomous Robots (2013), Software available at <http://octomap.github.com>.
- [LYT⁺16] X. Lu, J. Yao, J. Tu, K. Li, L. Li, and Y. Liu, *Pairwise linkage*

- for point cloud segmentation, ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences **III-3** (2016), 201–208.
- [Mag09] Martin Magnusson, *The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection*, Ph.D. thesis, Örebro universitet, 2009.
- [Mah36] Prasanta Chandra Mahalanobis, *On the generalized distance in statistics*, Proceedings of the National Institute of Sciences (Calcutta) **2** (1936), 49–55.
- [MCB11] J. Moras, V. Cherfaoui, and P. Bonnifait, *Credibilist occupancy grids for vehicle perception in dynamic environments*, 2011 IEEE International Conference on Robotics and Automation, May 2011, pp. 84–89.
- [PGK02] Mark Pauly, Markus Gross, and Leif P Kobbelt, *Efficient simplification of point-sampled surfaces*, Proceedings of the conference on Visualization'02, IEEE Computer Society, 2002, pp. 163–170.
- [PNDW98] D. Pagac, E. M. Nebot, and H. Durrant-Whyte, *An evidential approach to map-building for autonomous vehicles*, IEEE Transactions on Robotics and Automation **14** (1998), no. 4, 623–629.
- [SASL13] Jari P. Saarinen, Henrik Andreasson, Todor Stoyanov, and Achim J. Lilienthal, *3d normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments*, The International Journal of Robotics Research **32** (2013), no. 14, 1627–1644.
- [Sha76] Glenn Shafer, *A mathematical theory of evidence*, vol. 42, Princeton university press, 1976.
- [SLB18] Alexander Schaefer, Lukas Luft, and Wolfram Burgard, *Dct maps: Compact differentiable lidar maps based on the cosine transform*, IEEE Robotics and Automation Letters **3** (2018), no. 2, 1002–1009.
- [Sme94] Philippe Smets, *The transferable belief model*, Artif. Intell. **66** (1994), no. 2, 191–234.
- [Sme07] ———, *Analyzing the combination of conflicting belief functions*, Inf. Fusion **8** (2007), no. 4, 387–412.
- [SYM10] Nader Salman, Mariette Yvinec, and Quentin Mérigot, *Feature preserving mesh generation from 3d point clouds*, Computer graphics forum, vol. 29, Wiley Online Library, 2010, pp. 1623–1632.
- [TTWB14] Georg Tanzmeister, Julian Thomas, Dirk Wollherr, and Martin Buss, *Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation*, 2014 IEEE Intl. Conf. on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014, 2014, pp. 6090–6095.
- [TW17] G. Tanzmeister and D. Wollherr, *Evidential grid-based tracking and mapping*, IEEE Transactions on Intelligent Transportation Systems **18** (2017), no. 6, 1454–1467.



Appendix A

Contents of attached CD

Path	Description
/thesis.pdf	This thesis as PDF file.
/sw/	Folder containing all implemented and used source codes.
/videos/	Folder containing set of videos from experiments.