Comprehensive Automated 3D Urban Environment Modelling Using Terrestrial Laser Scanning Point Cloud

Pouria Babahajiani
Nokia Technologies
Tampere, Finland
pouria.babahajiani.ext@nokia.com

Lixin Fan
Nokia Technologies
Tampere, Finland
lixin.fan@nokia.com

Joni-Kristian Kämäräinen
Department of Signal Processing
Tampere University of Technology
Tampere, Finland
joni.kamarainen@tut.fi

Moncef Gabbouj
Department of Signal Processing
Tampere University of Technology
Tampere, Finland
moncef.gabbouj@tut.fi

Abstract

In this paper we present a novel street scene modelling framework, which takes advantage of 3D point cloud captured by a high definition LiDAR laser scanner. We propose an automatic and robust approach to detect, segment and classify urban objects from point clouds hence reconstructing a comprehensive 3D urban environment model. Our system first automatically segments grounds point cloud. Then building facades will be detected by using binary range image processing. Remained point cloud will be grouped into voxels and subsequently transformed into super voxels. Local 3D features are extracted from super voxels and classified by trained boosted decision trees with semantic classes e.g. tree, pedestrian, and car. Given labeled point cloud the proposed algorithm reconstructs the realistic model in two phases. Firstly building facades will be rendered by ShadVis algorithm. In the second step we apply a novel and fast method for fitting the solid predefined template mesh models to non-building labeled point cloud. The proposed method is evaluated both quantitatively and qualitatively on a challenging TLS NAVTEQ True databases.

1. Introduction

Analysis of 3D spaces comes from the demand to understand the environment surrounding us and to build more and more precise virtual representations of that space. 3D urban environment model is a digital representation of the earth´s surface and its related objects such as building, tree, vegetation, and some manmade feature belonging to the city area. There are various terms used for the 3D urban environment models such as “cyber town”, “virtual city”, or “digital city”. All of them are basically a computerized street model which contains the graphic representation of buildings and other objects in a 3D space. These 3D models are useful in variety of applications such as urban planning, crime prevention and control, virtual reality and robotic cars industry.

In this work, we propose a novel automatic 3D urban modeling approach which takes advantage of 3D geometrical features extracted from Light Detection And Ranging (LiDAR) point cloud. Since such 3D information is invariant to lighting and shadow, as a result, significantly more accurate results can be achieved.

While a laser scanning or LiDAR system provides a readily available solution for capturing spatial data in a fast, efficient and highly accurate way, the enormous volume of captured data often come with no semantic meanings. Some of these devices output several million data points per second. So efficient and fast methods are needed to filter the significant data out of these streams and high computing power is needed to post-process all this large amount of data. In despite of [1, 2], in this research we focus on a hybrid two stage voxel based classification to address the above mentioned challenges. Firstly, we adopt an unsupervised segmentation method to detect and remove dominant ground and buildings from other LiDAR data points, where these two dominant classes often correspond to the majority of point clouds. Secondly, after removing these two classes, we use a pre-trained boosted decision tree classifier to label local feature descriptors extracted from remaining vertical objects in the scene.

While most city modelling approaches are focused on facade modeling [3, 4, 5, 6], the focus of this work is automatic detection, segmentation and classification of whole urban objects. Given a point cloud containing one or more objects of interest and a set of labels corresponding to a set of models known to the system, the semantic segmentation system assigns correct labels to regions, or a set of regions, in the point cloud.

Given a labeled 3D point cloud with known position, orientation and shape, as well as the set of solid predefined street object templets, the 3D urban model will be generated.

1.1. Literature Review

While many image and video processing techniques for 2 dimensional object recognition have been proposed [7, 8], the accuracy of these systems is to some extent unsatisfactory because 2D image cues are sensitive to varying imaging conditions such as lighting, shadow and etc. In order to alleviate sensitiveness to different image

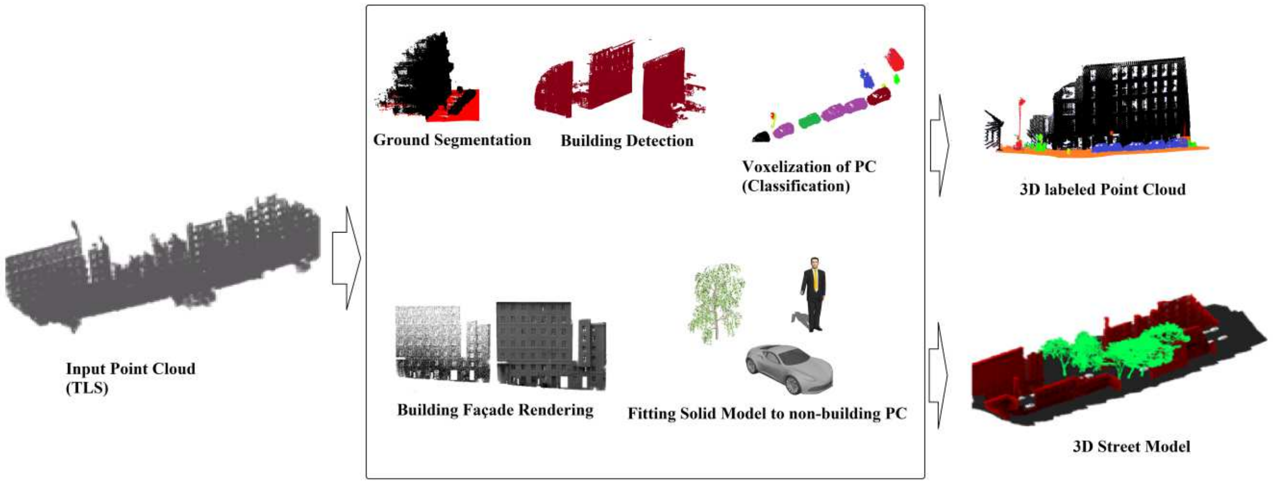
Ground Segmentation
Building Detection
Voxelization of PC (Classification)
3D labeled Point Cloud
Input Point Cloud (TLS)
Building Façade Rendering
Fitting Solid Model to non-building PC
3D Street Model

Fig 1. Framework of proposed mythology

capturing conditions, many efforts have been made to employ 3D scene features derived from single 2D images and thus achieving more accurate object recognition [9]. In the last decade, as the 3D sensors begun to spread and computing capacity for large scale 3D data processing became available, new methods and applications were born. Since such 3D information is invariant to lighting and shadow, as a result, significantly more accurate parsing results are achieved. 3D urban scene analysis had been often performed using 3D point cloud collected by airborne LiDAR for extracting vegetation and building structures [10]. Recently, classification of urban street objects using data obtained from mobile terrestrial systems has gained much interest because of the increasing demand of realistic 3D models for different objects common in urban era. A crucial processing step is the conversion of the laser scanner point cloud to a voxel data structure, which dramatically reduces the amount of data to process. Zhou and Yu (2012) present a voxel-based approach for object classification from TLS data [2].

Visual SLAM (Visual Simultaneous Localization and Mapping) has recently received a great attention within the robotics and vision communities which prepares consistent estimation of the 3D structure of the environment [13, 14] without considering the type and affordance of object in the scene.

Vosselman and Dijkman proposed a methodology and algorithm for 3D building model reconstruction from point clouds and ground plan [11]. They used the well-known Hough transform for the extraction of planar faces from the irregularly distributed point clouds. Ming et al., [12] investigated the methodology and algorithms for automatic generation of three dimensional photo realistic models from Lidar and image data. They implemented automatic 3D point cloud registration, automatic target recognition that is used for geo-referencing and automatic plane detection algorithm that is used for surface modeling, and texture mapping.

2. Proposed Methodology

It is a challenging task to directly extract objects from mobile LiDAR point cloud because of the noise in the data, huge data volume and movement of objects. We therefore take a hybrid two-stage approach to address the above mentioned challenges. Firstly, we adopt an unsupervised segmentation method to detect and remove dominant ground and buildings from other LiDAR data points, where these two dominant classes often correspond to the majority of point clouds. Secondly, after removing these two classes, we use a pre-trained boosted decision tree classifier to label local feature descriptors extracted from remaining vertical objects in the scene. This work shows that the combination of unsupervised segmentation and supervised classifiers provides a good trade-off between efficiency and accuracy. The output of classification phase is 3D labeled point cloud and each point is labeled with a predefined semantic classes such as building, tree, pedestrian and etc.

As photorealism cannot be achieved by using geometry cues alone, and because we aim to only use point cloud data (without image cues to decrease the procedure complexity) we present two post processing phases to enhance our proposed model visual quality. Firstly building facades will be rendered by ShadVis algorithm, then for non-building labeled point cloud we localize predefined template meshes to achieve more realistic model.

The contribution of this work are as follows:

- A complete scene parsing system is devised and experimentally validated using 3D urban scenes point cloud that have been gathered with LiDAR acquisition devices. The steps such as segmentation,

feature extraction, voxelization are generic and adaptable to solve object class recognition problems in different streets with varying landscape.
- Proposed two-stage (supervised and non-supervised) classification pipeline which requires only small amount of time for training.
- Propose to use novel geometric features leads to more robust classification results
- Generating textured façade of cities and localizing predefined templates for remained small objects such as car and pedestrian to enhance the visual quality of proposed model.

The framework of the proposed mythology is given in figure 1, in which 3D LiDAR point cloud is the inputs of the processing pipeline and parsing results are presented as 3D labeled point cloud. Final semi-photorealistic model of scene consisting textured 3D building facade and small template object are shown as post classification result.

A. Ground Segmentation

The aim of the first step is to remove points belonging to the scene ground including road and sidewalks, and as a result, the original point cloud are divided into ground and vertical object point clouds (Figure 2, A). Given a 3D point cloud of an urban street scene, the proposed approach starts by finding ground points by fitting a ground plane to the scene. This is because the ground connects almost all other objects and we will use a connect component based algorithm to over-segment the point clouds in the following step.

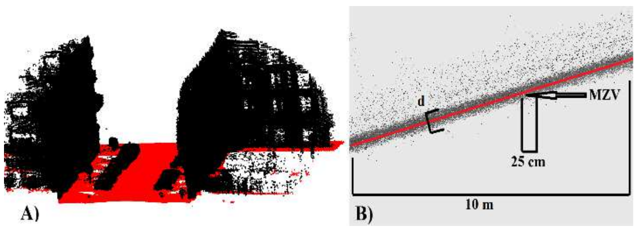

Fig 2. Ground Segmentation. A) Segmented ground and remained vertical objects point cloud are illustrated by red and black colour respectively. B) Sketch map of fitting plane to one tile

The plane RANSAC fitting method is used to approximate ground section of the scene. The RANSAC algorithm was developed by Fischler et al. [15] and is used to provide a more robust fitting of a model to input data in the presence of data outliers. Given a 3D point cloud of an urban street scene, the scene point cloud is first divided into sets of 10m×10m regular, non-overlapping tiles along the horizontal x–y plane. Then the following ground plane fitting method is repeatedly applied to each tile. We assume that ground points are of relatively small z values as compared to points belonging to other objects such as buildings or trees (see Fig 2).

The ground is not necessarily horizontal, yet we assume that there is a constant slope of the ground within each tile. Therefore, we first find the minimal-z-value (MZV) points within a multitude of 25cm×25cm grid cells at different locations. For each cell, neighboring points that are within a z-distance threshold from the MZV point are retained as candidate ground points. Subsequently, a RANSAC method is adopted to fit a plane to candidate ground points that are collected from all cells. Finally, 3D points that are within certain distance (d in Figure 2, B) from the fitted plane are considered as ground points of each tile. The approach is fully automatic and the change of two thresholds parameters do not lead to dramatic change in the results. On the other hand, the setting of grid cell size as 25cm×25cm maintains a good balance between accuracy and computational complexity.

B. Building Segmentation

Our method automatically extract building point cloud (e.g. doors, walls, facades, noisy scanned inner environment of building) based on two assumptions: a) building facades are sufficiently tall compare to the other structures in the street; and b) other non-building objects are located on the ground between two sides of street. As can be seen in figure 3, our method projects 3D point clouds to range images because they are convenient structures to process data. Range images are generated by projecting 3D points to horizontal x–y plane. In this way, several points are projected on the same range image pixel. We count the number of points that falls into each pixel and assign this number as a pixel intensity value. In addition, we select and store the maximal height among all projected points on the same pixel as height value. We define range images by making threshold and binarization of I, where I pixel value is defined as equation (1):

$$I_i = \frac{P_{intensity}}{\text{Max\_}P_{intensity}} + \frac{P_{\text{height}}}{\text{Max\_}P_{\text{height}}} \qquad (1)$$

Where I_i is grayscale range image pixel value, $P_{intensity}$ and P_{height} are intensity and height pixel value and Max_$P_{intensity}$ and Max_P_{height} represent the maximum intensity and height value over the grayscale image. On the range image, an interpolation is required in order to fill holes caused by occlusions, missing scan lines and LiDAR back projection scatter.

In the next step we use morphological operation (e.g. close and erode) to merge neighbouring point and filling holes in the binary range images (figure 3). The morphological interpolation does not create new regional maxima, furthermore it can fill holes of any size and no parameters are required. Then we extract contours to find boundaries of objects. In order to trace contours, Pavlidis contour-tracing algorithm [16] is proposed to identify each contour as a sequence of edge points. The resulting segments are checked on aspects such as size and diameters (height and width) to distinguish building from other objects. More specifically, equation (2) defines the geodesic elongation E(X), introduced by Lantuejoul and Maisonneuve (1984), of an object X, where S(X) is the area and L(X) is the geodesic diameter.

$$\mathrm{E}(X) = \frac{\pi\, L^2\,(X)}{4S(X)} \qquad (2)$$

Considering the sizes and shape of buildings, the extracted boundary will be eliminated if its size is less than a threshold. The resolution of range image is the only projection parameter during this point cloud alignment that should be chosen carefully. If each pixel in the range image cover large area in 3D space too many points would be projected as one pixel and fine details would not be preserved. On the other hand, selecting large pixel size compared to real world resolution leads to connectivity problems which would no longer justify the use of range images. In our experiment, a pixel corresponds to a square of size 0.05 m^2.

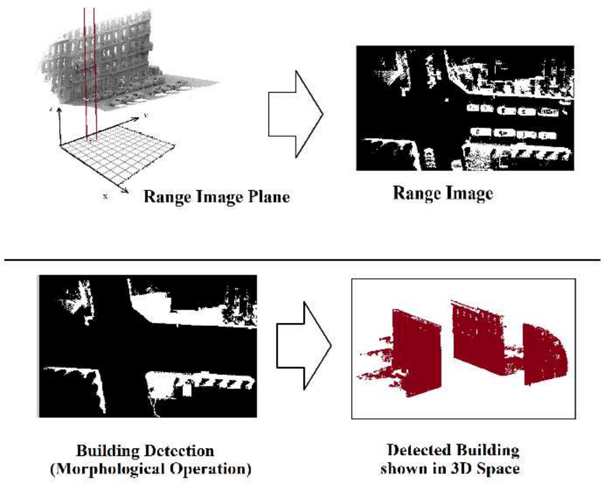
Range Image Plane
Range Image
Building Detection (Morphological Operation)
Detected Building shown in 3D Space

Fig 3. Building Segmentation

The 2D image scene is converted back to 3D space by extruding it orthogonally to the point cloud space. The x–y pixels coordinate of the binary image labeled as building facades are preserved as x–y coordinate of 3D point cloud (with open z value) labeled as building, and not considered in the remainder of our approach. Other points (negligible amount compare to the size of whole point cloud) are labeled as non-building class and will be later be classified as other classes e.g. car, tree, pedestrian and etc.

C. Voxel based classification

Although top view range image analysis generates a very fast segmentation result, there are a number of limitation to utilize it for the small vertical object such as pedestrian and cars. These limitations are overcome by using inner view (lateral) or ground based system in which, unlike top view the 3D data processing is done more precisely and the point view processing is closer to objects which provides a more detailed sampling of the objects. However, this leads to both advantages and disadvantages when processing the data. The disadvantage of this method includes the demand for more processing power required to handle the increased volume of 3D data.

According to voxel based segmentation, points which are merely a consequence of a discrete sampling of 3D objects are merged into voxels to represent enough discriminative features to label objects. 3D features such as intensity, area and normal angle are extracted based on these voxels. The voxel based classification method consists of three steps, voxelization of point cloud, merging of voxels into super-voxels and the supervised classification based on discriminative features extracted from super-voxels.

1) Voxelization of Point Cloud: In the voxelization step, an unorganized point cloud p is partitioned into small parts, called voxel v. The middle image in figure 4 illustrates an example of voxelization results, in which small vertical objects point cloud such as cars are broken into smaller partition. Different voxels are labelled with different colours. The aim of using voxelization is to reduce computation complexity and to form a higher level representation of point cloud scene. A collection of points is grouped together to form a variable size voxels. The criteria of including a new point pin into an existing voxel i is essentially determined by the crucial minimal distance threshold d_{th} which is defined as equation (3).

$$\min(\,\|P_{im} - P_{in}\|_2) \le d_{th}\,,\ \ 0 \le \mathrm{m,n} \le \mathrm{N},\ \ \mathrm{m} \ne \mathrm{n} \qquad (3)$$

Where p_{im} is an existing 3D point in voxel, p_{in} is a candidate point to merge to the voxel, i is the voxel index, d_{th} is the maximum distance between two point and N is the maximum point number of a voxel. If the condition is met, the new point is added and the process repeats until no more point that satisfies the condition is found. Equation (3) ensures that the distance between one point and its nearest neighbours belonging to the same voxel is less than d_{th}. Although the maximum voxel size is predefined, the actual voxel sizes depend on the maximum number of points in the voxel (N) and minimum distance between the neighbouring points.

2) *Super Voxelization*: For transformation of a voxel to super voxel we propose an algorithm to merge voxels via

region growing with respect to the following properties of voxels:

- If the *minimal geometrical distance*, D_{ij}, between two voxels is smaller than a given threshold, where D_{ij} is defined as equation (4):

$$D_{ij} = \min\left(\left\| P_{ik} - P_{jl} \right\|_2 \right), \ \mathrm{k} \in (1, \mathrm{m}), \mathrm{l} \in (1, \mathrm{n}) \qquad (4)$$

Where voxels v_i and v_j have m and n points respectively, and p_{ik} and p_{jl} are the 3D point belong to voxel v_i and v_j.

- If the angle between normal vectors of two voxels is smaller than a threshold: In this work, normal vector is calculated using PCA (Principal Component Analysis) [17]. The angle between two s-voxels is defined as angle between their normal vectors as equation (5):

$$\theta_{ij} = \arccos(< n_i \ , n_j >) \qquad (5)$$

Where n_i and n_j are normal vectors at v_i and v_j respectively. The proposed grouping algorithm merges the voxels by considering the geometrical distance ($D_{ij} < d_{th}$) and normal features of voxels ($\theta_{ij} < \theta_{th1}$). All these Voxelization steps then would be used in grouping these super-voxels (from now onwards referred to as s-voxels) into labeled objects.

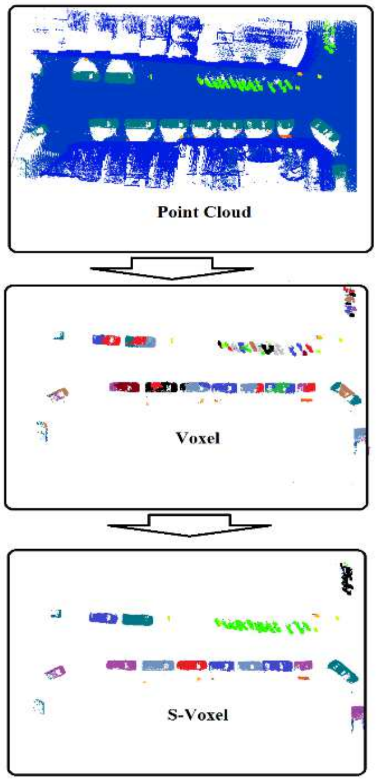

Fig 4. Voxelization of Point Cloud. from top to down: top view row point cloud, voxelization result of objects point cloud after removing ground and building, s-voxelization approach of point cloud

The advantage of this approach is that we can now use the reduced number of s-voxels instead of using thousands of points in the dataset, to obtain similar results for classification.

3) *Feature extraction:* For each s-voxel, seven main features are extracted to train the classifier.

Geometrical shape:

Projected bounding box has effective features due to the invariant dimension of objects. We extract four feature based on the projected bonding box to represent the geometry shape of objects.

-Area: the area of the bounding box is used for distinguishing large-scale objects and small ones.

-Edge ratio: the ratio of the long edge and short edge.

-Maximum edge: the maximum edge of bounding box.

-Covariance: is used to find relationships between points spreading along two largest edges.

Height above ground: Given a collection of 3D points with known geographic coordinates, the median height of all points is considered as the height feature of the s-voxel. The height information is independent of camera pose and is calculated by measuring the distance between points and the road ground.

Horizontal distance to center line of street: Following [1], we compute the horizontal distance of the each s-voxel to the centre line of street as second geographical feature. The street line is estimated by fitting a quadratic curve to the segmented ground.

Density: Some objects with porous structure such as fence and car with windows, have lower density of point cloud as compared to others such as trees and vegetation. Therefore, the number of 3D points in a s-voxel is used as a strong cue to distinguish different classes.

Intensity: Following [1], LiDAR systems provide not only positioning information but also reflectance property, referred to as intensity, of laser scanned objects. This intensity feature is used in our system, in combination with other features, to classify 3D points. More specifically, the median intensity of points in each s-voxel is used to train the classifier.

Normal angle: Following [18], we adopt a more accurate method to compute the surface normal by fitting a plane to the 3D points in each s-voxel. The surface normal is important properties of a geometric surface, and is frequently used to determine the orientation and general shape of objects.

Planarity: Patch planarity is defined as the average square distance of all 3D points from the best fitted plane computed by RANSAC algorithm. This feature is useful for distinguishing planar objects with smooth surface like cars form non planar ones such as trees.

4) *Classifier:* The Boosted decision tree [19] has demonstrated superior classification accuracy and robustness in many multi-class classification tasks. Acting as weaker learners, decision trees automatically select features that are relevant to the given classification problem. Given different weights of training samples, multiple trees are trained to minimize average classification errors. Subsequently, boosting is done by logistic regression version of Adaboost to achieve higher accuracy with multiple trees combined together. Each decision tree provides a partitioning of the data and outputs a confidence-weighted decision which is the class-conditional log-likelihood ratio for the current weighted distribution. In our experiments, we boost 10 decision trees each of which has 6 leaf nodes.

D. City model reconstruction using labeled point cloud

The 3D modelling methods are mainly categorized based on the input data techniques, one is photogrammetry (aerial, satellite and close range based model), and another one is the laser techniques (aerial, and terrestrial based model) which is the subject of this work. Most of the time the 3D model is generated using the photogrammetry techniques or when the model is built by laser scanners the registered image data is used to reconstruct textured 3D facades model. In these model implementation the 3D point cloud is registered with image data, automatic plane detection is used for surface modelling and texture mapping will be done using image data [4, 11]. Based on our knowledge, there is no literature review available on 3D city modelling just using 3D laser geometric data to generate realistic models without color imaging cues.

This subsection illustrates our approach that reconstructs the realistic model using input labeled point cloud in two phases. Firstly we use ShadVis algorithm [20] to render the building facades since the algorithm calculates the illumination of a point cloud (or vertices of a mesh) with the light coming from a theoretical hemisphere or sphere around the object. In the second step we apply a methods for fitting the solid predefined template models to non-building labeled point cloud.

1) *Building façade rendering:*

Recently, point-based geometry has become a very popular 3D object representation for geometry processing and graphics. The design of rendering tools using this point wise geometry representation is relatively straightforward. In terms of computer graphics research, this part has been inspired by previous research in point-based geometry and skydome/terrain rendering [20, 21].

We use the data structure of [22] for our point-based representation of the data since it allows flexible multi-level rendering with small overhead.

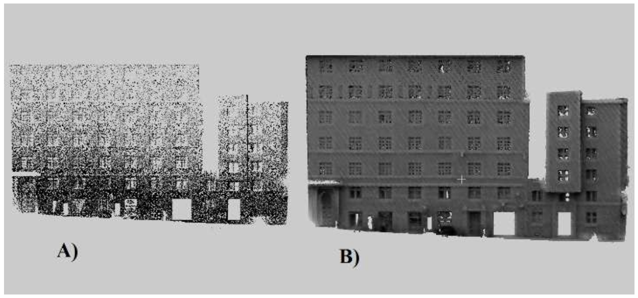

Fig 5. A) Input labelled point cloud representing building facades. B) The rendering result with the 3D skydome approach.

Even though graphics hardware rendering pipelines have been designed for polygons, the rendering of points is even easier than for polygons. So as [20], we use the adapted version algorithm of [22] for our purposes since it could only process surfaces and we have points as input. The accuracy of the result is concerned with the resolution of the 3D data. A snapshot from the 3D viewer of a facades with approximately 15 m length is shown in figure 5.

2) *Method for fitting solid model to non-building labelled point cloud*

In the last part we generate 3D building model using just 3D point cloud. Due to complexity of different urban scene most studies have been focused just on facades or building modelling [3, 4]. In this step we present the method based on fitting predefined template street view object to non-building label point cloud (such as car, tree, pedestrian and etc.) to devise a complete virtual 3D model of the urban scene.

The input of this approach is the classified labeled point cloud and the output is the solid meshes. We divided the street view object categories modelling into two subsets and adopt different object fitting approaches for them. The first subset is related to the object classes which their solid mesh structure orientation is not important and their object models will be completed based on only their position and dimension. These class model fitting include tree, pedestrian, sign symbol and etc. Unlike a lot of work which calculate the distance of a given points to the closet surface and use time consuming iterative procedure to fit the solid model into the point cloud or reconstructed surface [23], we propose a novel approach to solve this problem in a straightforward and computationally lightweight manner.

For each separated point cloud collection, the center and its boundaries will be calculated. Based on the size of the existing solid library meshes, we localize the best isodiametric meshes to the point cloud. As the object orientation is not important we only fit the mesh by stretching it to an appropriate size.

Object orientation should be considered for the second street view object categories such as car, bus, bike and general vehicle, therefore, we propose to fit the model based on the bounding box center matching. Similar to subset one the center of mesh and point cloud will be matched and then the corresponding model will be chosen from library based on the dimension of the vehicle bounding box. Then Iterative Closet Point (ICP) algorithm [23] is applied to automatically refine the registration of two entities.

All the fitting method assume that the correspondence between the points and the meshes will be successfully resolved during iterations of the fitting, unless after several iteration the orientation of the last vehicle will be considered. In figure 6 we show the result of fitting model to the point cloud. Notice that even with the difference in target and template type of the car (the solid template mesh is sedan and point cloud is hatchback) the pose is recovered accurately.

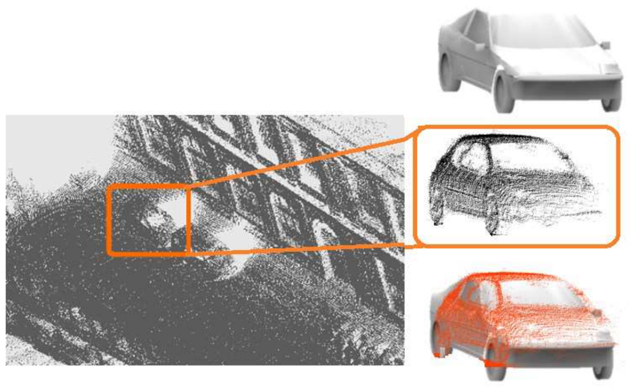
Fig 6. Fitting the mesh model into labeled vehicle point cloud. The top car is one of the solid model candidate. The middle car is the segmented point cloud which is labeled using supervised classifier. And the bottom image shows the fitting result.

The proposed method takes advantage of priori knowledge about urban scene environment and assumes that there are enough distance between different object in the street so that they are not connected.

3. Experimental Result

Our methodology is evaluated on three Mobile Laser Scanning (MLS) databases to get comparative results with state of the art: two NAVTAQ True [1] from Helsinki and Chicago, and rues Soufflot dataset from Paris [24].

As a general remark, our experiments starts with ground and building segmentation then we train boosted decision tree classifiers with sample 3D features extracted from training s-voxels. Subsequently we test the performance of the trained classifier using separated test samples. The accuracy of each test is evaluated by comparing the ground truth with the scene parsing results. Since our proposed modeling approach (fitting model to street view point cloud) has not been done before in existing work, we only compare the proposed classification approach both quantitatively and qualitatively.

We created and used labeled dataset of driving sequence from NAVTAQ True, provided by HERE, consists of best of sensors in positioning and LiDAR. The two NAVTAQ point cloud datasets contains more than 800 million points covering approximately 2.4 km altogether.7 semantic object classes are defined to label the LiDAR dataset: building, tree, car, sign symbol, person, fence and ground. It's noteworthy that several objects such as wall sign and wall light are considered as building facades.

The whole two NAVTAQ True datasets are mixed and divided into two portions: the training set, and the testing set. The 70% long of dataset are randomly selected and mixed for training of classifier and 30% remained long of point cloud is used for testing. Table 1 shows the quantities results achieved by our approach.

Table 1. Confusion matrix, NAVTAQ True datasets

	Tree	**Car**	**Sign**	**person**	**Fence**	**Ground**	**Building**
Tree	0.89	0.00	0.07	0.00	0.04	0.00	0.00
Car	0.03	0.95	0.00	0.00	0.02	0.00	0.00
Sign	0.17	0.00	0.72	0.11	0.00	0.00	0.00
person	0.02	0.00	0.20	0.78	0.00	0.00	0.00
Fence	0.03	0.00	0.00	0.00	0.85	0.00	0.12
Ground	0.00	0.00	0.00	0.00	0.00	0.98	0.02
Building	0.00	0.00	0.00	0.00	0.04	0.00	0.96

Mixing data from different cities poses serious challenges to the parsing pipeline, which is reflected by the decrease in the class average accuracy. Nevertheless, it seems our algorithm performs well on most per class accuracies, with the highest accuracy 96% achieved for the building and ground and the lowest 72% for sign. This low accuracy for small objects (e.g. person, sign) is mainly due to lack of sufficient training examples, which naturally lead to a less statistically significant labelling for objects in these classes. Moving objects are also hard to be reconstructed based solely on 3D data. As these objects (typically vehicles, people) are moving through the scene, which make them appear like a long-drawn shadow in the registered point cloud. The global accuracy is about 91 %. As it can be seen in the figure 7, successful point cloud classification and alignment have been done accurately.

It is noteworthy that our algorithms were initially developed to process NAVTAQ True data sets. One of the main advantages of our method is that it can be easily generalized to other datasets without any major