Online classification in 3D urban datasets based on hierarchical detection[*]

Thomas Flynn
Department of Computer Science
The Graduate Center of CUNY
tflynn@gradcenter.cuny.edu

Olympia Hadjiliadis
Department of Mathematics & Statistics
Hunter College & Graduate Ctr.
oh207@hunter.cuny.edu

Ioannis Stamos
Department of Computer Science
Hunter College & Graduate Ctr.
istamos@hunter.cuny.edu

International Conference on 3D Vision (3DV), October 2015, Lyon, France

Abstract

One of the most significant problems in the area of 3D range image processing is that of segmentation and classification from 3D laser range data, especially in real-time. In this work we introduce a novel multi-layer approach to the classification of 3D laser scan data. In particular, we build a hierarchical framework of online detection and identification procedures drawn from sequential analysis namely the CUSUM (Cumulative Sum) and SPRT (Sequential Probability Ratio Test), both of which are low complexity algorithms. Each layer of algorithms builds upon the decisions made at the previous stage thus providing a robust framework of online decision making. In our new framework we are not only able to classify in coarse classes such as vertical, horizontal and/or vegetation but to also identify objects characterized by more subtle or gradual changes such as curbs or steps. Moreover, our new multi-layer approach combines information across scanlines and results in more accurate decision making. We perform experiments in complex urban scenes and provide quantitative results.

1. Introduction

The photorealistic modeling of large-scale scenes, such as urban structures, has received significant attention in recent years [9]. Outdoor urban environments are complex due to the variability of objects (such as buildings, people, cars, street level structures, roadways, curbs, etc. - see Fig. 1), occlusions, partial views, multiple resolutions and noise. On the other hand, indoor scenes face similar challenges due to clutter, significant occlusions as well as variability of materials (such as glass or metal). One of the most significant problems is that of segmentation and classification from 3D laser range data. Time-of-flight laser scanners acquire 3D points sequentially. In that set-up, it makes sense to develop classification techniques that operate on-the-fly, as data is being acquired. Even in the case of other scanner technologies, real-time classification through fast sequential techniques is desirable. Important applications include augmented reality, and scene understanding for robotics.

In this work we develop a system for the sequential classification of 3D points captured by a range scanner, by introducing a novel hierarchical framework which consists of various low-complexity change-detection algorithms. A change detector is defined as a method that distinguishes between two states from a stream of observations. In our previous work [15], it was shown how the basic format of the CUSUM/SPRT change-detection algorithms (to be defined in Sec. 4) could be used to build a toolbox of useful sequential classifiers. In the current paper, we introduce a novel hierarchical classification framework by extending this basic scheme. This results in a robust and efficient set of routines for online classification.

Hierarchical information processing is ubiquitous and has proven very useful in artificial intelligence, particularly in the field of neural networks. With this in mind, we explore the idea of building a hierarchy of change detectors. Within this framework, while the first layer of processing may involve detecting changes in distribution directly on the *sensor output*, subsequent layers of the hierarchy receive input in the form of *alarms* emitted by change detectors lower in the hierarchy. We are classifying points acquired by a 3D range scanner into one of the categories $\{vegetation, vertical, horizontal, curb\}$. Note that the location of curbs can be used for identifying drivable terrain and can serve as a cue for the identification of other objects of interest. This procedure can be adapted for detection of various step-like changes (such as stairs for instance).

In the next section we review related work. Then in Sec. 3 we provide an overview of our system. Sec. 4 describes the basic CUSUM and SPRT hypothesis testing procedures. Secs. 5 through 7 go into more detail about the components of our system, and in Sec. 8 we discuss the complexity of the algorithm. Finally, in Sec. 9 we present experimental results on a collection of 3D urban scenes.

[*] This work has been supported in part by NSF grants IIS- 0915971 and CCF-0916452, DMS-ATD #1222526 as well as the CUNY Collaborative Research (R20), CUNY Central, Hunter Presidential and PSC-CUNY.

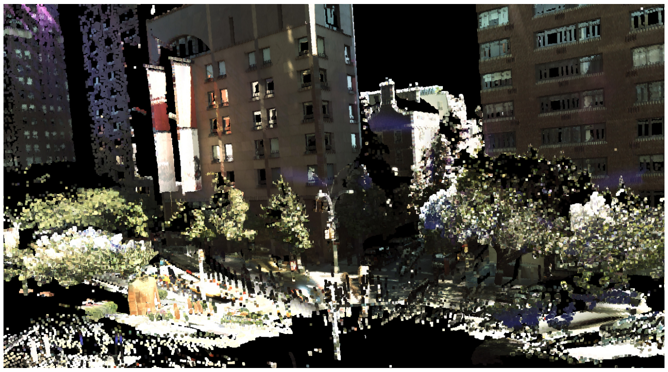


Figure 1. Part of 3D range scan of a complex urban scene.

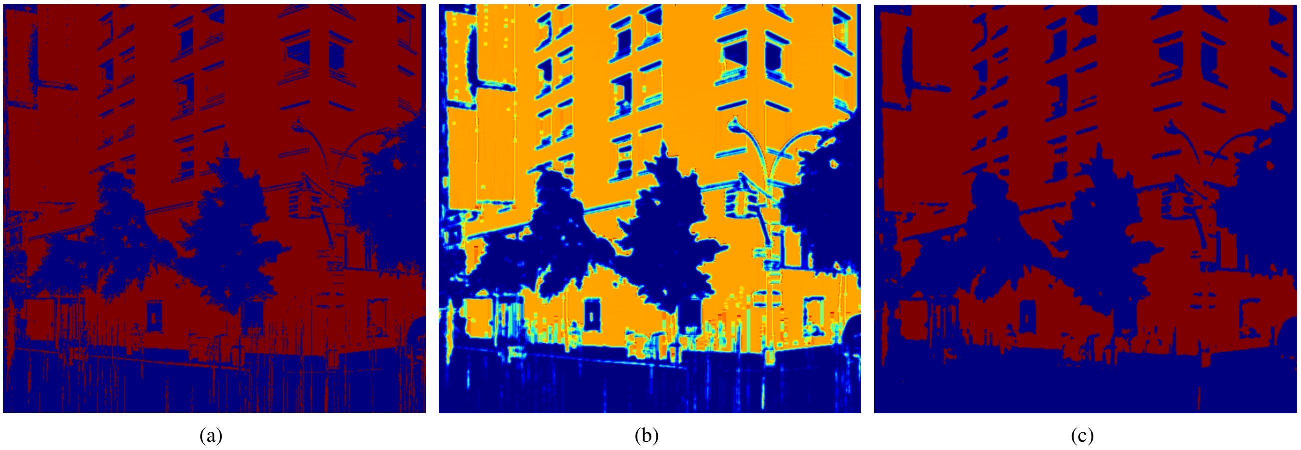


Figure 3. Close up view of second layer processing for the vertical class. From an input classification $\delta_{ij}^{(1)}$ (a), the SB_{IJ} statistic (b) is computed in an online-fashion and then fed into a second layer CUSUM, giving the out classification in (c).

2. Related work

Classifying different types of objects has received a great deal of attention in recent years [3, 10, 2, 15]. Most work in 3D vision is based on the detection of features and the use of machine learning techniques for classification in an offline fashion. In [4], urban objects are segmented and classified by comparing them to annotated examples using shape and contextual features. In the early work of [6], spin-image features are used for the recognition of objects in cluttered scenes. In [3], the Maximum Margin Associative Markov Networks for point cloud labeling were introduced. [19] extended it with more classifiers, while [8] generalized it to high-order Markov Random Fields (MRFs) using the Functional Boosting method. [20] presents a multi-stage inference procedure that uses point cloud statistics to learn relational/contextual information between classes, while [12] presents message-passing methods for classification. The technique is applied to point cloud classification and 3D surface estimation from single images. The work of [17, 18] involves the recognition based on keypoints, while [14] provides a comparison of various 3D keypoint descriptors (such as [13]) within the context of classification.

However, these techniques assume that the data becomes available all at once, as opposed to sequentially. Online detection techniques have been used in 3D Computer Vision but mainly in the context of a moving car [7, 16], where rough object types need to be identified for determination of drivable vs. non-drivable surfaces. Our earlier work [5] constructs online detection techniques to classify points in one of three classes, namely horizontal, vertical and vegetation by exploiting the sequential nature in which data points are acquired by the 3D laser range-scanner. Yet, the techniques developed therein use only a first layer of online classification algorithms which are limited to information provided within each scanline and thus fail to combine vital information contained across scanlines.

Considering a comparison to more standard approaches, the online nature of our algorithm means that as soon as data begins streaming in, the classification of each new point re-

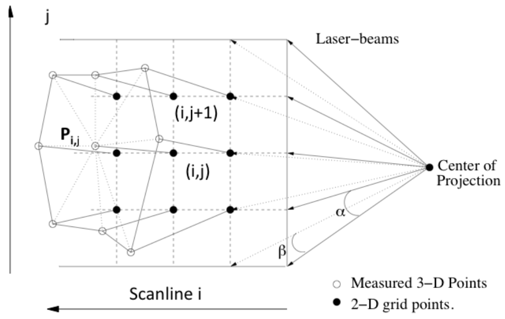


Figure 2. Description of laser-scanning mechanism. A set of laser-beams are emitted in a sequential fashion. The angle between the beams in the vertical direction is α and in the horizontal β. The device measures the distance between the center of projection and the surface point along the beam direction. Each point is acquired sequentially in a raster-scanning order (i.e. first column then second column, etc., within a column first row, second row, etc.).

quires a constant amount of time, comparable to the acquisition time. This is in contrast to methods based on, e.g. MRFs, where inference is a global operation involving optimization. At the same time, we are able to achieve some benefits of MRFs-namely the ability to enforce smoothness. In MRFs this is achieved through pairwise or higher order potential functions while here we use aggregation (Sec. 7).

Finally, hierarchical systems involving change detection have been explored for instance in [1]. In that work, a two level change detection procedure is described, in which the rise of an alarm from one change detector activates another change detector that confirms or rejects the alarm raised by the first. Also, intrusion detection systems have employed change detection schemes at various stages of processing. In these systems, the output of change detection elements serve as input to other processing modules. However, a novel feature of the present work is that we use the output of one change detector as the input to *another change detector*. This is reminiscent to the fact that neural networks are not just hierarchical information processing systems, but they are hierarchies of homogeneous components, wherein the output of one can serve as input to another.

3. System overview

Our system consists of three stages. The first layer produces a *coarse classification* of points into one of the classes $\{vegetation, vertical, horizontal\}$. This is done via a sequential approach using the CUSUM and SPRT procedures, which are described in the next section. The second layer implements *pattern detection* on this classification. This is relevant for object classes that can be characterized by a certain pattern of alarms produced by change detectors. Curbs are an example of this; they can be identified by a pattern of the form "horizontal → vertical → horizontal". This is described in more detail in Sec. 6. The final layer is *aggregation*, which eliminates noise in the first layer classification by taking into account observations from larger neighborhoods. This is described in Sec. 7. Although each stage serves a different purpose, a single methodology is used in their design: (1) identify useful summary statistics on the input and (2) apply sequential identification and/or detection routines in the proper combination on these statistics.

Time-of-flight range scanners acquire 3D points in a sequential fashion, whereas triangulation-based scanners can acquire a larger number of points at once. In both cases, the 3D points are organized in a structured grid manner, as shown in Fig. 2. As a result, we acquire a grid of 3D points $P_{i,j} = [x_{ij}, y_{ij}, z_{ij}]$, for $i = 1, \ldots, M$ and $j = 1, \ldots, N$.

Let us first introduce the collection of major object classes $\mathcal{C} = \{C_k\}$, $k = 1, \ldots, K$ that are present in a scene. We then focus on the specific class C_k for a fixed k which we denote by C and define the indicator variables

$$1_{ij} = \begin{cases} 1 & \text{iff } P_{i,j} \text{ is of class } C \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

and the decision variables

$$\delta_{ij}^{(r)} = \begin{cases} 1 & \text{iff } P_{i,j} \text{ is decided to be of class } C \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where r denotes the stage in which a decision is taken with $r = 1, \ldots, R$ and $i = 1, \ldots, M$, $j = 1, \ldots, N$. In fact, the decision rules at stage $r + 1$ are built on top of statistics of decisions made at stage r. This enables us to compute a coarse classification into major classes at first and to subsequently use this initial classification to identify patterns that characterize more detailed objects. The final layer of classification is used to robustify decisions on object types by combining information in adjacent scanlines.

To be more specific, we list the different classifications we intend to achieve in the following three layers:

Layer 1 We first classify the 3D points in one of three classes $C_1 : Horizontal$, $C_2 : Vertical$, $C_3 : Vegetation$, using information within each scanline.

Layer 2 We then further classify the 3D points into given patterns. One of these patterns could be a transition from horizontal to vertical and then back to horizontal. Based on the height of the vertical region such a pattern would signify the existence of a curb. Note that other similar patterns (such as steps) can be detected following the same paradigm.

Layer 3 Finally, we aggregate the above classifications, by combining results from neighboring scanlines.

It is important to mention that $\delta_{ij}^{(r)}$ are not necessarily determined at the time the point $P_{i,j}$ is scanned. In the examples developed below, $\delta_{ij}^{(R)}$ is determined based on a statistic whose value depends on the decisions $\delta_{ij}^{(r)}$, for $r = 1, \ldots, R-1$ computed after a window of size $n \times n$ (which requires at least n scanlines) becomes available. Using information across scanlines results in, as expected, more robust and accurate decisions.

4. Online detection and identification algorithms

This work builds off well known algorithms from sequential analysis. Many online classification tasks can be phrased as the problem of detecting the time at which the process generating observations switches from one statistical mechanism to another. The CUSUM (Cumulative SUM) change-detection procedure is one such change detector, and is known to satisfy a number of optimality conditions [11]. Closely related is the SPRT (Sequential Probability Ratio Test), a sequential hypothesis testing scheme [11]. We use various combinations of these two procedures to classify points, as described below.

We briefly review the CUSUM and SPRT. More details can be found in [11]. To describe the CUSUM and SPRT rules we begin by considering the summary statistic or observation $V_{i,j}$ for $j = 1, 2, \ldots, N$ associated with each point $P_{i,j}$ for every fixed scanline i. There are two hypotheses at hand regarding the distribution of these summary statistics, namely

$$\begin{aligned} H_0: \quad & V_{i,j} \text{ i.i.d. } \sim f_0(x), \quad j = 1, 2, \ldots, N \\ & \qquad \text{versus} \\ H_1: \quad & V_{i,j} \text{ i.i.d. } \sim f_1(x), \quad j = 1, 2, \ldots, N \end{aligned} \tag{3}$$

and two cases stemming from them:

1. The observations follow H_0 for $j = 1, 2, \ldots, v-1$ and H_1 for $j = v, v+1, \ldots, N$, where $v > 0$ is the change point.

2. The observations follow either H_0 or H_1 for all $j = 1, 2, \ldots, N$.

In the first case the objective is to stop and declare an alarm as soon as possible. The CUSUM stopping rule is known for its optimal property of balancing the trade off between a small detection delay for any given tolerance on the mean time to the first false alarm. It is defined as

$$T(h) = \inf\{j > 0; u_j \geq h\} \tag{4}$$

where $u_0 = 0$; $u_j := \max\{0, u_{j-1} + g(V_{i,j})\}$ and

$$g(x) = \ln \frac{f_1(x)}{f_0(x)} \tag{5}$$

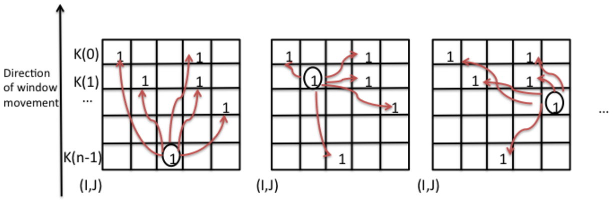


Figure 4. A demonstration of the aggregate statistic SB_{IJ} computation for a window of size 5×5. The places in the window with $\delta_{ij}^{(1)} = 1$ are labeled as 1. For each location (i, j) with $\delta_{ij}^{(1)} = 1$ we calculate the Euclidean distances between P_{ij} and every other point P_{kl} with $\delta_{kl}^{(1)} = 1$ in the same window. For a demonstration we show three such locations as circles in the figure.

The threshold h is then determined by the user's tolerance on the mean time to the first false alarm, namely

$$E_{H_0}[T(h)] = \gamma. \tag{6}$$

In the second case the objective is to decide between H_0 and H_1 by devising a sequential decision rule (s.d.r.) which is a pair of a stopping rule S and a decision variable Δ_S the latter of which takes values in $\{0, 1\}$ corresponding to hypotheses H_0 and H_1 respectively. The idea is to devise an s.d.r. that minimizes the number of observations required to make a decision subject to given type I $(P_{H_0}(\Delta_S = 1) = \alpha)$ and type II $(P_{H_1}(\Delta_S = 0) = \beta)$ error probabilities. It is known that the SPRT S enjoys such an optimality property [11] where

$$S = \inf\{n \geq 0; g(V_{i,j}) \notin (a, b)\}$$

with $g(V_{i,j})$ as in (5) and the constants a and b are defined through $b \approx \log \frac{1-\beta}{\alpha}$, $a \approx \log \frac{\beta}{1-\alpha}$. The decision variable takes the value 1 if exit takes place on b and 0 if exit takes place on a. Note that the implementation of the these algorithms only requires the ability to evaluate the ratio of the likelihoods under each hypothesis.

5. First layer: Coarse classification

The first stage of processing in our system is a preliminary classification of points into one of several classes. This is performed as in [15]. As an example of the first layer of algorithms, we consider classifying point $P_{i,j}$ as vertical, generating the decision variables $\delta_{ij}^{(1)}$. A summary statistic which can be used for this can be calculated from the sequence $D_{i,j} = P_{i,j+1} - P_{i,j}$ defined within each scanline i. We define the sequence $V_{i,j}$ $j = 1, \ldots, N$ as the angles between the vector $D_{i,j}$ with the pre-determined **z** axis. See Fig. 5a for a visualization of this statistic. The resulting sequence can then be modeled as Gaussian variables $\{V_{i,j}\}$, $j = 1, 2, \ldots, N$ with a different mean under each of the hypotheses H_0 and H_1 respectively. More specifically, horizontal surfaces are characterized by $E_{H_0}[V_{i,j}] = \mu_0 = 90°$

while vertical surfaces by $E_{H_1}[V_{i,j}] = \mu_1 = 0°$. We assume that the scanner is initially placed on a horizontal surface and as such the CUSUM stopping rule (4) is designed to detect a change from the mean μ_0 to the mean μ_1 above. Under the Gaussian distributional assumption the form of $g(x)$ in (5) becomes $g(x) = -\left[x - \frac{\mu_1+\mu_0}{2}\right]$ which ensures the proper functionality of both CUSUM and the subsequent SPRT rules.

To concretize the final values of our decision variables $\delta_{ij}^{(1)}$, suppose that the CUSUM stopping rule T of (4) goes off at observation L. We then run the SPRT and suppose it stops at observation $L + Q$ after exit on the right (i.e. $\Delta_S = 1$). Then $\delta_{ij}^{(1)}$ are set to 1 for all j, such that $L \leq j < L + Q$. Also $\delta_{ij}^{(1)}$ is set to 0 for all j with $j < L$. The SPRT is repeated starting from observation $L + Q + 1$. If $\Delta_S = 1$ at observation $L + Q + X$ then $\delta_{ij}^{(1)}$ are set to 1 for all $L + Q + 1 \leq j < L + Q + X$ and to 0 if $\Delta_S = 0$. The SPRT is repeated from observation $L + Q + X + 1$. The reader may consult [15, 5] for more details.

6. Second layer: pattern detection

Information about where curbs are in a 3D scene can be very useful low level information for tasks further along a pipeline. For instance it can be used to help detect drivable terrain for autonomous driving or mapping applications. Information about curbs can also be used as contextual information when classifying building facades or ground/sidewalk areas.

Our curb detector is a component of second layer processing, meaning that curbs are detected by a combination of CUSUM/SPRT procedures whose inputs are a signal derived from a combination of CUSUM/SPRT outputs at the first layer of processing described in Sec. 5. The form of the statistics used for curb detection are motivated by a few simple observations about curbs and the area surrounding them. Firstly, there is a gradual change in the height of points where a curb occurs, going from very close to the ground to the sidewalk. Secondly, the curb appears as a vertical segment of short length between two horizontal sections (before and after the curb). Therefore, we use both the height of points above the ground (determined by their z-coordinate), and the output of the vertical classifier $\delta_{ij}^{(1)}$ to construct the statistic used for curb detection.

In particular, the summary statistics $V_{i,j}$, $j = 1, 2, \ldots, N$ will be the z-coordinate of each point $P_{i,j}$. We then make distributional assumptions for $V_{i,j}$ according to the hypotheses (3). In particular we set $f_0(x) = \phi(\frac{x-\mu_g}{\sigma})$ and $f_1(x) = \epsilon_j \phi(\frac{x-\mu_g}{\sigma}) + (1 - \epsilon_j)\phi(\frac{x-\mu_{pw}}{\sigma})$, where μ_g stands for the mean of the z-coordinate on the ground and μ_{pw} stands for the mean of the z-coordinate on the pedestrian walk. Note that $\phi(\frac{x-\mu}{\sigma})$ is the Gaussian kernel with mean μ and variance σ. Both parameters μ_g and μ_{pw} can be estimated from prior data as can σ which is usually estimated as a fraction of the difference between μ_g and μ_{pw}. The parameters ϵ_j $j = 1, 2, \ldots, N$ use the classification variables $\delta_{ij}^{(1)}$ as follows:

1. $\epsilon_j = 0$ for all $j = 1, \ldots, v$, where $v = \min\{m; \delta_{im}^{(1)} = 1\}$,
2. $\epsilon_j = k\eta$ for $j = v + k$, $k = 1, 2, \ldots, u$, where $u = \min\{m > v; \delta_{im}^{(1)} = 0\}$ and η is set to $\frac{1}{\text{\# points in pattern of interest}}$.

We can estimate the # of points in pattern of interest using the distance from the scanner, the scanning resolution, and from prior data.

The intuition behind the above is as follows. The distribution $f_0(x)$ describes the state of the world before the appearance of the curb. Before a curb, the z-coordinates should be around the mean z-coordinate for the ground. Then, there is a gradual change to the z-coordinate of the pedestrian walk. The 3D points that are part of this gradual change should have been classified as vertical from the classifier of the previous level. The distribution $f_1(x)$ is a mixture model that describes just that gradual change, by taking into account the classification of the previous level in the form of the parameters ϵ_j that act as weights.

The above discussion largely assumes that the curb is not running parallel to the scanning direction. For curbs parallel to the scanning direction, performing the sequential processing in the direction across scanlines would be sufficient.

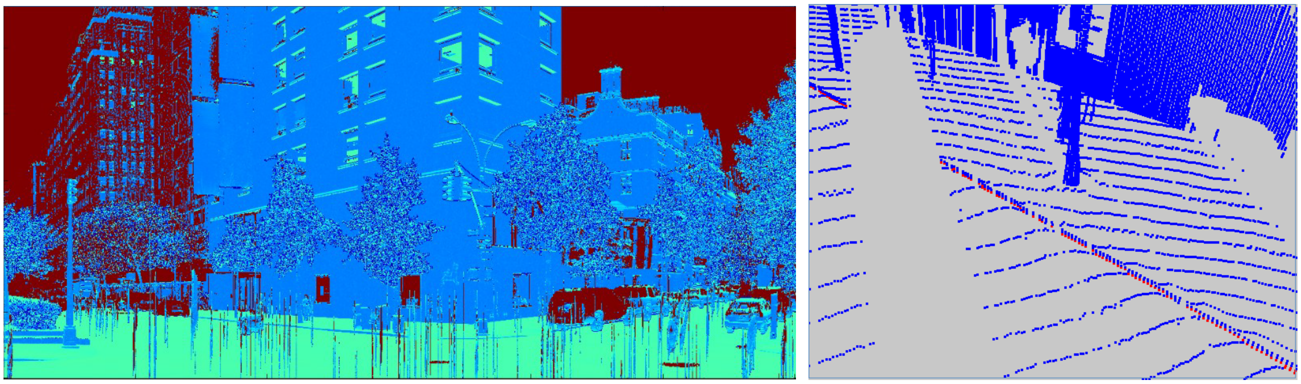
Using the above distributional assumptions we use the algorithm described in the second paragraph of Sec. 5 to determine the decision variables $\delta_{ij}^{(2)}$ (i.e. the curb classification). Fig. 5b shows a curb detected by this procedure.

7. Final layer : aggregation

The first or second layer of algorithms produce a local decision per point, which may be noisy. The goal of the final layer is to aggregate the results of the first (or second) layer algorithms, by focusing on aggregating first (or second) layer decisions $\delta_{ij}^{(1)}$ (or $\delta_{ij}^{(2)}$). In the next two sections we consider the aggregate statistic and the final layer CUSUM on the statistic that we use to achieve this.

7.1. Aggregate statistic

We consider windows of size $n \times n$ that run sequentially on the result of the previous layer (for example on the $\delta_{ij}^{(1)}$ shown in Figs. 3(a),6(a)). We can consider various aggregation functions on this window. One proposed function is described below. At each position (i, j) in the $n \times n$ window for which $\delta_{ij}^{(1)} = 1$ has been provided from the first layer, we can calculate the distances in the Euclidean



(a)

(b)

Figure 5. (a) Signed angles summary statistic. The color of each point is related to its signed angle (red denotes no input). (b) Result of curb algorithm. Red points indicate detected curb.

space between point $P_{i,j}$ and every other point $P_{k,l}$ with a $\delta_{kl}^{(1)} = 1$ in the same window. A window is represented by its lower left corner (I, J) and its size n. Within each window we calculate the following aggregate scalar as the summary statistic:

$$SB_{IJ} = \sum_{i=I}^{I+(n-1)} \sum_{j=J}^{J+(n-1)} K(I+(n-1)-i)\, \delta_{ij}^{(1)}\, D_{i,j}, \quad (7)$$

where

$$D_{i,j} = \sum_{\substack{k=I \\ k \neq i}}^{I+(n-1)} \sum_{\substack{l=J \\ l \neq j}}^{J+(n-1)} \delta_{kl}^{(1)}\, G_{I,J}(||P_{i,j} - P_{k,l}||)$$

and $I = 1, \ldots, M-n+1, J = 1, \ldots, N-n+1$. $G_{I,J}()$ is the Gaussian kernel $\phi\left(\frac{x-\mu}{\sigma}\right)$ with mean μ equal to the sample mean of all pairwise distances between points $P_{i,j}$ with $\delta_{ij}^{(1)} = 1$ in the window, and variance σ^2 the sample variance of the same pairwise distances. $K()$ is a Gaussian kernel of 0 mean and of standard deviation $\sigma = n/5$. For a given window size n, the values $K(n-1), \ldots, K(0)$ need to be computed once. The kernel values $K(i)$ act as weights in order to favor points closer to the direction of movement of the window. Note that when the distance between two points is close to the sample mean, the Gaussian kernel $G_{I,J}()$ will be close to its maximum value. For large pairwise distances the Gaussian kernel will be close to zero. Therefore, in windows with many points that are tightly concentrated in space, SB_{IJ} will be large. In windows with a few points that are away from each other in space, SB_{IJ} would be small. In addition, we favor distances from points closer to the window movement direction (via the use of $K()$), because we want to accelerate the increase of SB_{IJ} as the window is moving from an area of low concentration of points to an area of high concentration.

The fact that the mean of $G_{I,J}()$ is the sample mean and variance of pairwise distances in the window is important. Areas of further away objects are sampled less densely from areas of close objects. That means that the pairwise distances in windows of further away objects will be larger, than the ones from close by objects. The use of the sample mean and variance removes the bias generated by denser sampling of the closer to the scanner objects.

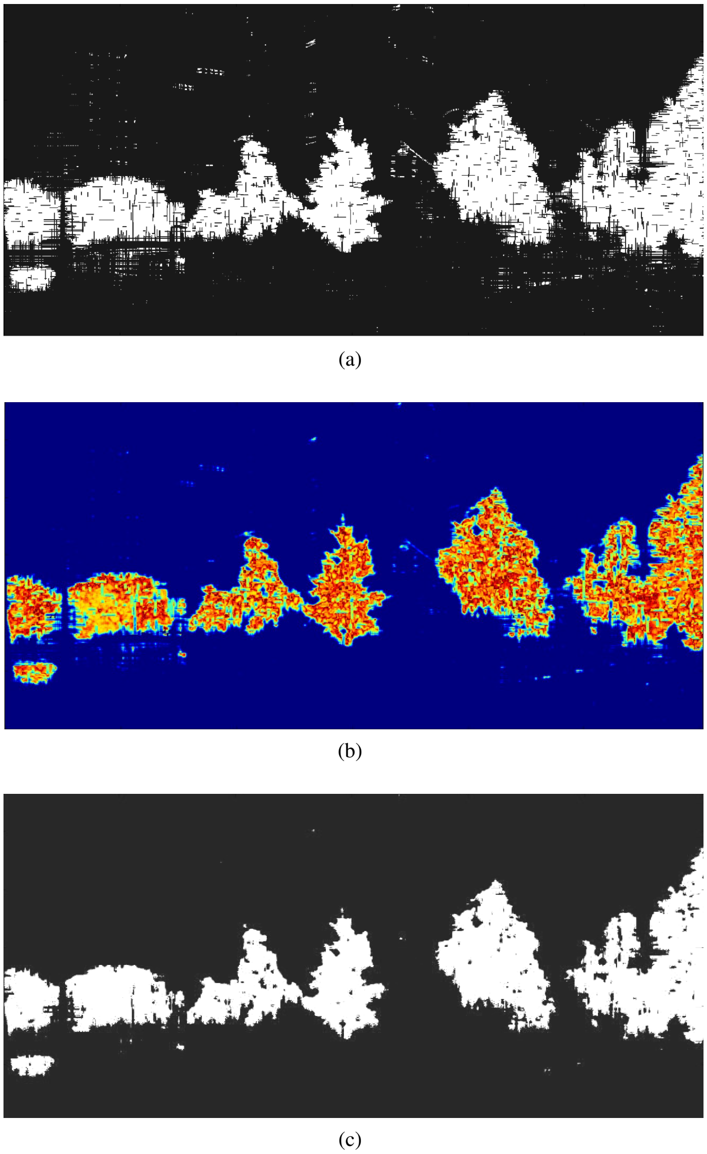
An illustration of how the statistic is computed is given in Fig. 4. The statistic requires many pairwise distances to be computed; these can be cached for efficiency. Figs. 3 and 6 show the SB_{IJ} statistic and the refined classifications, in the case of vertical and vegetation respectively. In each case we find that noise has been reduced significantly.

7.2. Final layer CUSUM

As stated above, a CUSUM procedure is run on the SB statistic. Assume that we have calculated the decision variables $\delta_{ij}^{(1)}$ for the vegetation class. We now calculate the SB_{IJ} statistic on an $n \times n$ moving window both in a vertical (i.e. fix scanline I and vary $J = 1, \ldots, N-n+1$) and in a horizontal direction (i.e. fix J and vary $I = 1, \ldots, M-n+1$). Initially the procedure assumes that the category is absent and is thus detecting a change from the hypothesis $H_0 = No\ vegetation$ to $H_1 = Vegetation\ present$. Once the change has been detected, a separate test takes over which detects the change from H_1 back to H_0. Specifically, focusing on scanline I, a change from H_0 to H_1 is detected as $T_c = \min\{k; d_k > h\}$ where

$$d_J = \max\left\{0, d_{J-1} + \left(SB_{IJ} - \frac{\mu_0 + \mu_1}{2}\right)\right\},$$

for $J = 1, 2, \ldots$. The parameters to be chosen in the above are the threshold h and the means μ_0, μ_1, which should correspond to the mean of the SB_{IJ} statistic under H_0 and H_1



(a)

(b)

(c)

Figure 6. Close up view of final layer processing for vegetation classification. From an input classification $\delta_{ij}^{(1)}$ (a), the SB_{IJ} statistic (b) is computed in an online-fashion and then fed into a final layer CUSUM, giving the final classification in (c). Note that noise that appears in (a) has been eliminated in (c).

respectively. Likewise, a change from H_1 to H_0 is detected using $T_c' = \min\{k; d_k' > h'\}$ where

$$d_J' = \max\left\{0, d_{J-1}' - \left(SB_{IJ} - \frac{\mu_0 + \mu_1}{2}\right)\right\},$$

for $J = 1, 2, \ldots$.

For the parameter μ_0, it is easy to see that under H_0 the mean of the SB_{IJ} is 0, and thus we set $\mu_0 = 0$. The value of μ_1 is set to 100 which is an approximation of the value of the SB_{IJ} in regions with a high density of $\delta_{ij}^{(1)} = 1$. The thresholds h and h' are set to 50 for the purpose of our experiments in light of (6). It is important to note here that the CUSUM alarm times T_c and T_c' are robust to the choice of μ_1 whose only strict restriction is $\mu_1 > 0$. In fact it is the difference in μ_1 and μ_0 that matters.

8. Complexity Analysis

Our method is online and can run in real-time. Signed angle, CUSUM, and SPRT computations are $O(1)$ per 3D point and require $O(1)$ memory. The complexity of the aggregate statistic (Sec. 7) depends on the window size n selected, and n is to be kept small in order not to over-smooth and for real-time processing. In the worst-case scenario computation of each $SB_{I,J}$ (Eq. 7) requires calculation of all pairwise distances and it costs $O(n^3)$ per point. This is due to the fact that distances calculated for $SB_{I,J}$ can be carried over to $SB_{I,J+1}$ (or $SB_{I+1,J}$). The Gaussian weights K used in the computation are constant and can be pre-computed. Since n is small the computation of the aggregate statistic is fast, and memory requirements are $O(1)$.

9. Experiments

We tested the algorithms described above on a large dataset of urban scenes. The data was captured using a Leica ScanStation 2 scanner that obtains points in the order depicted in Fig. 2. We performed two sets of experiments. First we tested the ability of the system to classify points into one of the three classes $\{vegetation, vertical, horizontal\}$. This experiment was performed on a set of 9 scans, totalling about 6.4 million points. The scenes contained buildings, cars, light poles, streets, sidewalks and smaller items such as fire hydrants, garbage cans, etc., typical to urban settings. Ground truth data was obtained using custom labelling software. The results are shown in Tables 1 and 2 below. For each class we report the precision, recall, and F1 score. Note that the precision increases, while recall decreases (except of the horizontal class, where the opposite is true). The available ground truth data for curbs was smaller, and we performed a smaller experiment($\approx 400,000$ points) where points were classified into one of $\{vegetation, vertical, horizontal, curb\}$. The precision-recall results are shown in Table 3. Fig 7 shows a sample of ground truth data and the classification output by our system. The curb detection is very robust.

These numbers demonstrate a reduction of the number of false positives due to aggregation, without sacrificing too many true positives. For example the false positives for vegetation were reduced significantly from around $360,000$ points to $207,000$ points. One aspect not captured in Tables 1 and 2 is that second layer processing significantly improves the cohesion of detected objects. This can be seen by comparing Figs. 9 and 7.1. We suspect that poor Horizontal numbers can be due to the difficulty in labeling by hand horizontal points in numerous windows in large facades.

Metric	Vegetation	Vertical	Horizontal
Precision	0.704	0.971	0.773
Recall	0.835	0.806	0.958
F1	0.764	0.881	0.856

Table 1. Results using only first layer (coarse) classification (9 scans)

Metric	Vegetation	Vertical	Horizontal
Precision	0.793	0.991	0.738
Recall	0.811	0.797	0.99
F1	0.802	0.883	0.845

Table 2. Classification results after aggregation (9 scans)

Metric	Vegetation	Vertical	Horizontal	Curb
Precision	0.872	0.981	0.827	0.883
Recall	0.936	0.885	0.965	0.902

Table 3. Results using only first layer (coarse) classification (one scan)

There are a number of constants which need to be set in our system. The parameters of the CUSUM on the SB_{IJ} statistic are discussed in the last paragraph of the previous section. Also, computing the statistic requires the selection of a window size n, for which we used $n = 5$ in the experiments. The curb detector from Sec. 6 requires the mean of the sidewalk, μ_{pw} (taken to be $\mu_g + 0.1$), and the multiplier η (we used 0.2 since the approximate number of points scanned on a curb given the position of our scanner is about 5), used to calculate the sequences ϵ_j defined in that section. The CUSUM threshold used for the curbs was $h = 0.0001$ again in light of (6). Note that the subtlety of the change in the statistics induced by curbs leads to a choice of a small threshold though the algorithm is robust to its exact value.

We also report on the timing of our algorithm. For one of our typical scans, the scanner attempted to acquire $681,400$ points. The processing of this scan took around 12 minutes in a 2.2 GHz laptop, leading to an average of 1.05 milliseconds per point. This is almost the time of 1ms required for the acquisition of a single point with this last-generation scanner.

10. Conclusion

In this work we sought to demonstrate that the methodology of sequential detection can serve as a unifying principle in the design of hierarchical systems for classification. We found that added layers of sequential processing, notably the aggregation step of Sec. 7 significantly improved the performance of our system. Our methodology should also be applicable to other sequential tasks outside of 3D vision. Future work could possibly explore this, as well as the integration of machine learning methods.

References

[1] C. Alippi, G. Boracchi, and M. Roveri. A hierarchical, nonparametric, sequential change-detection test. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2889–2896, July 2011.

[2] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena. Contextually guided semantic labeling and search for three-

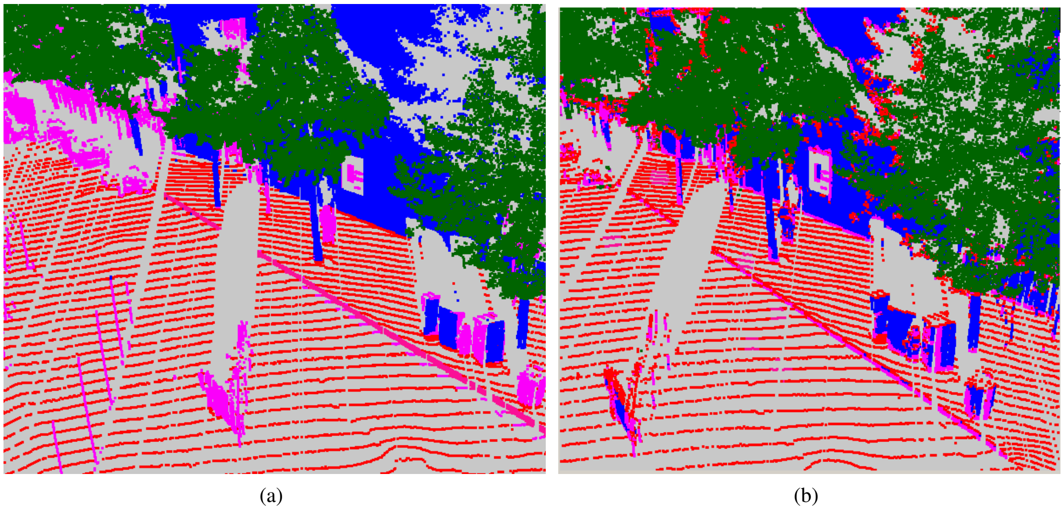


Figure 7. Visualization of classification. Fig. (a) shows ground truth data for the classes ground (red), vegetation (green) and vertical (blue). Purple denotes no ground-truth label. Fig. (b) shows the final classification of our system.

dimensional point clouds. *The International Journal of Robotics Research*, 2012.

[3] D. Anguelov, B. Taskarf, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of Markov random fields for segmentation of 3D scan data. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005.

[4] A. Golovinskiy and T. Funkhouser. Min-cut based segmentation of point clouds. *Computer Vision Workshops (ICCV Workshops)*, 2009.

[5] O. Hadjiliadis and I. Stamos. Sequential classification in point clouds of urban scenes. In *5th International Symposium on 3D Data Processing, Visualization and Transmission*, Paris, France, May 2010.

[6] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *Pattern Analysis and Machine Intelligence*, (July), 1999.

[7] M. Montemerlo and etal. Junior: The stanford entry in the urban challenge. *Springer Tracts in Advanced Robotics*, 56:91–123, 2009.

[8] D. Munoz, J. Bagnell, and M. Hebert. Co-inference for Multi-modal Scene Analysis. In *European Conference on Computer Vision (ECCV)*, 2012.

[9] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Gool, and W. Purgathofer. A survey of urban reconstruction. In *Computer Graphics Forum*, volume 32, pages 146–177. Wiley Online Library, 2013.

[10] A. Patterson IV, P. Mordohai, and K. Daniilidis. Object detection from large-scale 3d datasets using bottom-up and top-down descriptors. In *Computer Vision–ECCV 2008*, pages 553–566. Springer, 2008.

[11] H. V. Poor and O. Hadjiliadis. *Quickest Detection*. Cambridge University Press, 2008.

[12] S. Ross, D. Munoz, M. Hebert, and J. A. Bagnell. Learning message-passing inference machines for structured prediction. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2737–2744, June 2011.

[13] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D recognition and pose using the Viewpoint Feature Histogram. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162, Oct. 2010.

[14] S. Salti, F. Tombari, and L. D. Stefano. A Performance Evaluation of 3D Keypoint Detectors. *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 236–243, May 2011.

[15] I. Stamos, O. Hadjiliadis, H. Zhang, and T. Flynn. Online algorithms for classification of urban objects in 3D point clouds. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 332–339. IEEE, 2012.

[16] S. Thrun and etal. Stanley, the robot that won the DARPA grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006.

[17] F. Tombari and L. Di Stefano. 3D Data Segmentation by Local Classification and Markov Random Fields. *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, pages 212–219, May 2011.

[18] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *European Conference on Computer Vision (ECCV)*, pages 347–360, 2010.

[19] R. Triebel and R. Schmidt. Instance-based AMN classification for improved object recognition in 2D and 3D laser range data. *IJCAI'07 Proceedings of the 20th international joint conference on Artifical intelligence*, pages 2225–2230, 2007.

[20] X. Xiong, D. Munoz, J. A. Bagnell, and M. Hebert. 3-D Scene Analysis via Sequenced Predictions over Points and Regions. In *ICRA*, 2011.