Classification of Outdoor 3D Lidar Data Based on Unsupervised Gaussian Mixture Models

Artur Maligo, Simon Lacroix

Classification of Outdoor 3D Lidar Data Based on Unsupervised Gaussian Mixture Models

Artur Maligo[1] and Simon Lacroix[2]

***Abstract*— 3D point clouds acquired with lidars are an important source of data for the classification of outdoor environments by autonomous terrestrial robots. We propose here a two-layer classification system. The first layer consists of a Gaussian mixture model, issued from unsupervised training, which defines a large set of data-oriented classes. The second layer consists of a supervised, manual grouping of the unsupervised classes into a smaller set of task-oriented classes. Because it uses unsupervised learning at its core, the system does not require any manual labelling of datasets. We evaluate the system on two datasets of different nature, and the results show its capacity to adapt to different data while providing classes which are exploitable in a target task.**

I. INTRODUCTION

Perception is a key requirement for terrestrial autonomous mobile robots operating in outdoor environments. In particular, the processing of 3D point clouds acquired with lidars enable robots to build environment models, on which are based the solutions to tasks such as traversability analysis [1], object recognition [2], scan registration, place recognition [3] and others involving data association. Semantic models, in this context, are especially interesting because they encode qualitative information, and thus provide to a robot the ability to reason at a higher level of abstraction.

At the core of a semantic modelling system, lies the capacity to classify the sensor observations acquired from a target scene [4]. The challenges faced arise, firstly, from the difficulty of modelling the variability encountered in outdoor environments, which contain elements of all shapes and scales, possibly cluttered together [5], [6]. Secondly, the manner in which scene elements are sampled by a lidar depends on their position relatively to the sensor, on occlusions, and on the characteristics of the lidar.

Although supervised learning can be employed in the classification [7], [8], it is not scalable with respect to the amount and complexity of the concerned data, due to the necessity of manual labelling by a human domain expert. A different approach is to apply unsupervised learning, which overcomes this necessity because it is able to detect the classes that are naturally represented in the data.

In this work, we propose a two-layer classification model which mainly relies on unsupervised learning. The first, intermediary layer consists of a Gaussian Mixture Model (GMM) trained in an unsupervised manner, defining a set of intermediary classes which is a fine-partitioned representation of the environment. The second and final layer consists of a grouping of the intermediary classes into a set of final classes that are exploitable in the considered target task. The grouping process is done by an expert, therefore being supervised, but remains simpler than the manual labelling of a dataset.

The two-layer model is able to separate the factors that influence the classification. Data-oriented factors, that is the sensor and environment properties contained in the data, are abstracted by the intermediary layer. The final layer, in turn, introduces the task-oriented factors, that is, it gives classes a semantic interpretation.

A normal application of our classification process consists in the data acquisition, followed by the unsupervised training and supervised grouping of a few different models to be tested, followed by a qualitative, visual inspection of the results, and concluded by the choice of the model which performed the best. The output model is thus a predictive model and can be used to classify new data.

We evaluate our method on two datasets acquired with different lidars and possessing different characteristics. We evaluate it quantitatively with the first set, and qualitatively with both sets. Our system delivers consistent results, illustrating its generic nature and capacity of detecting the relevant classes in a scene.

In section II, we review the related work. Section III presents the main concepts of our approach and provides details about its implementation. We then introduce the experimental setup in section IV, and evaluate our system in section V. The paper ends in section VI with a short discussion and pointers to future work.

II. RELATED WORK

A. Classification Element

The classification element is the element being classified. It can be a 3D point, a segment, a voxel, or another structure. The choice of the classification element is linked to the type of environment model to be built.

In pointwise classification, classification is applied directly to 3D points [5], [7], [9]. Only local information, that is information about the neighbourhood of a point, is used for classification. Therefore, no assumptions regarding the segmentation of the points are made, making this approach agnostic with respect to shapes.

Some approaches apply segmentation on the points and then use the segments as classification targets [2], [6], [10].

[1]Artur Maligo is with CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France and Univ de Toulouse, INSA, LAAS, F-31400 Toulouse, France artur.maligo@laas.fr

[2]Simon Lacroix is with CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France and Univ de Toulouse, LAAS, F-31400 Toulouse, France simon.lacroix@laas.fr

This permits the use of global information in the classification, *i.e.* information about the whole object. This approach allows for a richer description of objects, but it introduces the constraint of dealing with all the variety of shapes.

There are methods that consider a more specific form of segments: voxels [8], [11]. In these works, points are grouped into voxels of adaptive sizes, then a subsequent segmentation step is applied, resulting in super-voxels, which are the targets of classification.

We believe that pointwise classification has the advantage of not biasing the classification by introducing an arbitrary segmentation, be it a fixed discretization or a data-centred segmentation. Moreover, the first layer of our model is oriented towards representing the basic shape patterns in the environment. Hence we opt for this approach.

B. Learning

Supervised learning is frequently applied in 3D data classification. A comparison is presented in [5]. [9] uses linear classifiers, [2] uses a SVM, [7] uses a GMM and [8], [12] use a CRF. The GMM used in [7] is supervised, with a fixed number of Gaussian components for each class.

Supervised learning has the disadvantage of requiring manual labelling of the dataset, hardly applicable if the amount of data is large, or if the process of labelling is complex. Moreover, in a difficult problem, where the considered classes are not well represented in the feature space, solutions tend to rely on more complex models, although these might not provide the most natural way of approaching the problem.

The use of unsupervised learning is relatively less common. The work of [6] presents a method where 3D points are segmented and the resulting segments are used for the unsupervised discovery of classes. [13] uses online clustering to incrementally learn classes, based on segments of a triangular mesh. [14], an unsupervised method based on range image features is used to generate a set of words, which are in turn used to replace similar regions of a map to compress its size. The work of [3] applies k-means clustering to range image features in order to assist in the place recognition problem.

In unsupervised learning, no classes are imposed, which leaves the model free to find the patterns that can be encountered in the data. A disadvantage is that the resulting classes do not possess an immediate semantic interpretation, and for this reason are not readily useful.

Our approach aims at avoiding manual labelling and at finding a model which naturally adapts to the data. We choose for this an unsupervised GMM. The works closest to ours are [6], [13], but they stop at the class discovery stage. The use of a final layer, in our approach, makes it possible to add a semantic interpretation to the discovered classes.

C. Scale

Classification is performed on a feature vector, resulting from a feature extraction process [15]. When 3D data is considered, scale arises as an essential factor in this process.

Considering pointwise classification, a standard method is, given a target point, to take all points lying inside a spherical support region centred around it, and use these in the feature computation [3], [5], [7]. Given that a sphere radius is specified, the resulting feature only provides information about the point neighbourhood on the specified scale. This method is not efficient when the classes present in the environment are characterized by different scales.

To overcome the problem mentioned above, multi-scale methods have been proposed. In [16], an adaptive process is performed: the radius of the support region is chosen based on the shape of the neighbourhood. This method is however computationally expensive.

Another multi-scale approach was proposed in [9]. In this work, multiple spherical support regions, with different radii, are used simultaneously for feature extraction. The resulting vector is a combination of the feature values extracted at the different radii, and thus encodes how the shape of the point's neighbourhood is perceived at different scales.

[17] presents a hierarchical approach for dealing with multiple scales. A point cloud is firstly analysed as a whole. If it is not considered flat according to their criterion, it is divided in halfs, following a 2D grid model. These halfs, which are 2D cells, are then submitted to the same analysis. This procedure continues in a recursive manner, and the division terminates if a cell is considered flat or if it has reached a minimum size.

Works applying segment classification deal with the scale problem in an implicit way, because segments assume different sizes depending on the object being segmented [2], [6], [8], [11].

In this work, besides considering a single spherical support region for feature extraction, we also explore the method of using multiple regions simultaneously, found in [9] and discussed above. In the first, single-scale case, what our model learns is the classes existing at the given scale. In the second, multi-scale case, the model learns the classes that present a consistent, specific pattern over the scales.

III. APPROACH

Our approach relies on the proposed two-layer classification model. We perform pointwise classification, such that a point, associated with its support region, or neighbourhood, is the element being classified. In the multi-scale case, a point is characterized by multiple neighbourhoods.

The classification model is composed by two layers. The intermediary layer consists of an unsupervised GMM. This layer provides the intermediary classes. The final layer consists of a grouping of the intermediary classes into the final classes, which are the output of the system.

A. Intermediary Layer: Unsupervised GMM

The intermediary classification layer is a GMM. Through feature extraction, a 3D point belonging to a point cloud is associated with a point $\mathbf{x}$ in the feature space. A GMM represents the distribution over $\mathbf{x}$ by introducing a latent

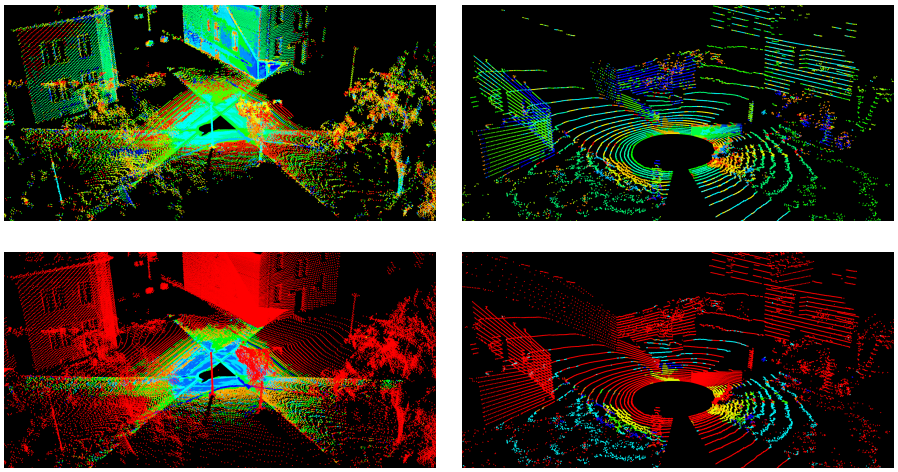

Fig. 1. Grouping principle. Left: a scan from the Freiburg dataset. Right: one from the Caylus dataset. Top: intermediary classes, colours encode the class. Bottom: intermediary classes that may be grouped into a final class. In the Freiburg scan, the final class is *ground*, in the Caylus scan, it is *grass*. Colours: red encodes the classes that won't be grouped, other colours encode the classes that will.

variable $\mathbf{c}$ [15]:

$$p(\mathbf{x}) = \sum_{\mathbf{c}} p(\mathbf{c})p(\mathbf{x}|\mathbf{c}) = \sum_{n=1}^{N_C} \pi_n \mathcal{N}(\mathbf{x}|\mu_n, \mathbf{\Sigma}_n). \quad (1)$$

Each possible value of $\mathbf{c}$ is a component of the model and corresponds to a class. N_C is the number of classes. The parameters of the model are, for each class, the mixing coefficient π_n, the mean μ_n and the covariance matrix $\mathbf{\Sigma}_n$.

Inference is done by computing the posterior distribution:

$$p(c_n = 1|\mathbf{x}) = \frac{\pi_n \mathcal{N}(\mathbf{x}|\mu_n, \mathbf{\Sigma}_n)}{\sum_{i=1}^{N_C} \pi_i \mathcal{N}(\mathbf{x}|\mu_i, \mathbf{\Sigma}_i)}. \quad (2)$$

Inference is followed by the decision step, where the class that obtained the highest posterior probability is assigned to the input.

Given a training set, the model parameters are found with the unsupervised Expectation-Maximization (EM) method. The complexity of the model is determined by the number of classes, or Gaussian components, employed in the GMM. By increasing the number of components of the model, it is ideally possible to model arbitrary decision boundaries in the feature space. This layer is currently implemented using scikit-learn [18].

B. Final Layer: Supervised Grouping

Figure 1 illustrates the principle of the grouping, the final layer. This layer is a mapping of the intermediary classes into a smaller set of final classes. Each final class has a semantic interpretation in the context of the target task. However, not all the intermediary classes can be exploited. Some of them correspond to objects of different nature, and thus cannot be grouped into a meaningful final class. In this case, the class is marked as *unknown*. *unknown* points do not contribute to the resulting classification. In a way, this procedure is analogous to a decision process, where we refrain from classifying a point, if a given certainty criterion is not met.

The grouping is determined during the training phase. This step is done in a supervised manner, by a human expert. Overall, it consists in examining the results of the intermediary classification, and assigning to each intermediary class a final class, or the class *unknown*. To perform this task, a graphical interface is required. In our work, we found that the visualization tool ParaView [19] provided all the desired functions.

C. Feature Extraction

The feature extraction process is performed pointwise. In the single-scale case, it takes into account a target point and the points in its spherical neighbourhood of radius r. In the multi-scale case, it takes into account multiple spherical neighbourhoods, determined by a set of radii $R = \{r_1, ..., r_{N_R}\}$, N_R being the number of radii. Three values are computed for each scale, resulting in a feature space dimension of 3, if single-scale, or $3N_R$, if multi-scale.

The input point cloud is expressed in the sensor reference frame. As we show further, for the first two feature values this is enough, but for the third one we require the transformation to the world reference frame. Thus, the inputs of feature extraction are actually the point cloud and the corresponding sensor-to-world transformation.

The three feature values result from a Principal Component Analysis (PCA) operation applied to a point's neighbourhood. The works of [7], [9], [12], [17] also base the feature extraction on PCA, and indeed our features are heavily inspired on, and similar to theirs, but with some differences. We use PCL [20] to implement the neighbours' search and PCA.

Let $\lambda_1 > \lambda_2 > \lambda_3$ be the eigenvalues output by PCA, and v_1, v_2 and v_3 the eigenvectors. As done in [9], the shape of the points distribution can be encoded by the following values, which we take as the first two feature values:

$$x_1 = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \lambda_3}, x_2 = \frac{\lambda_2}{\lambda_1 + \lambda_2 + \lambda_3}. \quad (3)$$

Interpreting PCA as a plane fitting operation [21], the eigenvector v_3, associated to the smallest eigenvalue λ_3, represents

an estimation of the surface normal. It is possible to use the local-to-global transformation to transform this vector into the global reference frame, resulting in the global normal $\mathbf{n} = [n_x \ n_y \ n_z]^T$. The third feature value, x_3, is then given by the z coordinate:

$$x_3 = n_z. \tag{4}$$

The feature extraction process includes a feature standardizing step. Based on the training set, we extract the mean μ_i and the standard deviation σ_i along each feature dimension i. Then, once all features are computed for a point, standardizing is applied for every dimension:

$$x_i = \frac{x_i - \mu_i}{\sigma_i}. \tag{5}$$

The final feature vector is given by $\mathbf{x} = [x_1 \ x_2 \ x_3]^T$ in the single-scale case, which becomes $\mathbf{x} = [\mathbf{x}_1^T \ ... \ \mathbf{x}_{N_R}^T]^T$ in the multi-scale case, $\mathbf{x}_j$ being the feature values computed at radius r_j. In this latter case, we proceed as in [9] and copy the feature values from a higher scale, if available, to the values at missing lower scales.

In the 3D space, it makes no sense applying PCA on a set with less than four points, because such points will always be collinear or coplanar. Four points, on the contrary, can either be collinear, coplanar, or none of both, and thus can characterize arbitrary 3D shapes. Thus, during feature extraction we leave out points with less than three neighbours. Such points are then also excluded from the classification. This situation occurs with higher frequency in the furthest regions of scans, where the laser sampling is more sparse. In turn, an interesting consequence is that isolated outliers are naturally left out of classification.

IV. EXPERIMENTAL SETUP

To test our approach, experiments with two outdoor datasets of different nature were performed. The first one is the Freiburg public dataset [3], for which we have ground-truth, made available in [5]. With this dataset, we can evaluate the system quantitatively, using standard metrics of classification performance. The second one is a dataset acquired with our own robot and sensor setup, for which there is no ground-truth. In this case, the results are evaluated qualitatively.

For each dataset, it is necessary to define two sets: a training set and a validation set. The training set is used for finding the GMM's parameters and for the grouping. The validation set is used for the evaluation. Both sets are composed by scans selected from the dataset. This section presents the experimental setups and explains how our system is expected to behave when handling the data from each dataset.

A. Freiburg Dataset

This is a public dataset, containing scans acquired with a SICK LMS lidar on board of a wheeled robot. The place is a university campus, including buildings of different types, roads, and many artificial elements like bikes, columns and street lamps. It also contains some grass areas, trees of different shapes, vegetation and, in some of the scans, people. The ground-truth considers 20 different classes, offering a fine distinction between the elements in the set.

For the acquisition, the sensor was moved using a pan-tilt unit. The point clouds are denser, if compared to the ones of the Caylus dataset. Here the scans are the result of the fusion of individual and overlapping shots, producing point densities that vary according to the area in the scan. This effect is noticeable on the ground and on the walls, for example, which can appear differently in areas where the scans overlap.

With this dataset, we test two instances of our system: a single-scale one and a multi-scale one. For a quantitative evaluation, we show the confusion matrices, and we calculate the metrics of precision, recall, F_1 and accuracy. The first three ones assess performance classwise, but by averaging F_1 over the classes we have a total F_1 score. These metrics are calculated on a validation set which does not contain any data from the training set.

B. Caylus Dataset

This dataset contains scans acquired with a Velodyne HDL-32E lidar, mounted on top of an unmanned ground vehicle. The Velodyne sensor is a mobile scanning sensor, acquiring data with a 360° horizontal field of view. The place is a countryside village, and contains earth paths, some slopes, low and high grass, trees, bushes and other vegetation, but also artificial elements like asphalted roads, buildings, road signs and some abandoned vehicles. The operator of the robot can be seen in some scans.

This dataset has a great deal of variety regarding natural elements, like terrain and vegetation. There are lots of clutter, and objects appear grouped in different ways through the dataset. As an example, tree trunks are sometimes isolated, sometimes have stones nearby, other times vegetation. The other challenging point of this dataset is the sparsity of the Velodyne sensor's sampling pattern. The ground, for instance, exhibits very different aspects in regions near to and far from the sensor, due to the dramatic decrease of the point density in function of the distance.

With this dataset, we evaluate the system qualitatively, due to the lack of ground-truth. We only test a single-scale instance of the system. The qualitative evaluation is done by visual inspection of scans in a validation set. This is an example of dataset for which ground-truth is difficult to produce, due to two factors: the sampling sparsity and the presence of more natural, non-structured elements. This case represents what would be a real application of our approach, starting from a dataset with no ground-truth, and ending with an inspection of the final classification results provided by the system.

C. Choice of Parameters

The first factors influencing the system's performance are the training set and the validation set. The important point is to keep the training and validation sets different in order to evaluate the generalization properties of the model. This

means selecting validation scans as far as possible from the training scans. At the same time, both sets should contain instances of all the relevant classes, in quantities as balanced as possible.

The other parameters affecting the behaviour of the system are N_C, the number of classes in the GMM, and r or R, the radius or set of radii of the support regions, respectively. N_C must be chosen so that the GMM is able to produce a fine enough set of classes, so that these can be grouped afterwards. The radii parameters determine the scales at which the model operates.

For the Freiburg tests, we manually choose five scans for the training set and grouping, and another five for the validation set, following the guidelines mentioned previously. Examining the validation results of some models, we select the two ones that performed best, one single-scale and another multi-scale. Both models have $N_C = 50$. The first one has $r = 50cm$, and the second has $R = \{0.2, 0.4, 0.6, 0.8, 1.0\}$. We present the results for these models on the next section.

For the Caylus dataset, twelve scans compose the training set, and from these two representative scans were selected for the grouping. Following the validation of some models, the values empirically selected are $N_C = 30$ and $r = 25cm$.

V. EVALUATION

A. Freiburg Results

For the Freiburg dataset, both the single-scale and the multi-scale models result in a set of four final classes: *ground*, *wall-building*, *pole-trunk-people*, and *foliage-vegetation-bicycle*. A fifth class, *unknown*, denotes the intermediary classes that could not be grouped into any meaningful class. For the purposes of evaluation, the original ground-truth classes are all grouped into the same four classes. Figure 2 shows the results for two scans from the validation set, and table I shows the quantitative results.

Among the final classes, *ground* has a single semantic meaning. *wall-building* includes small walls, buildings, as well as small shrubs, which all possess vertical planar shapes. *pole-trunk-people* groups three elements of different nature, but with a similar geometrical shape: vertical, linear, cylindrical. It is thus natural to find them under the same class. Moreover, there are very few instances of people in the data, so it is unprobable that the model would find such a specific class. A similar analysis applies to the class *foliage-vegetation-bicycle*. Bicycles end up under this label because they appear to the sensor as a scattered 3D shape, just as foliage and vegetation do.

Concerning the performance, the results for the classes *pole-trunk-people* and *foliage-vegetation-bicycle* are the lowest. This happens in part due to the rarity of these classes in the dataset, compared to the other two, and in part due to their geometrical characteristics. Poles and trunks look frequently similar to the corner of buildings and to the divisions between windows, and the bigger trunks look like walls. *foliage-vegetation-bicycle* suffers from the same difficulties, because it is confused with any element in the scene which has a marked 3D shape, like corners and prominent features of buildings.

The accuracies are the same for both models, 0.82, but looking at the precision, recall and F_1 scores, we can see that the multi-scale model performs better. An increase in the F_1 score from 0.72 to 0.76, in particular, is important because F_1 is a demanding score. The better performance of the multi-scale model is expected, since using more scales should lead to a richer distinction of the patterns in the data. It should be noted, however, that the single-scale model is not that far behind, which means that using a radius of $50cm$ already leads to a good representation of the classes.

Overall, we note that both models achieve a F_1 score above 0.70 and an accuracy above 0.80. Visually, the results are consistent, as it can be seen in figure 2. The precisions are high for all classes. The limitation of the system lies clearly in the recall. For every class, the highest number of false negatives is found under the *unknown* class. Thus, the impossibility of using all the intermediary classes provided by the GMM, leaving some of them unknown, is the main reason behind the lower recall scores.

We argue, in turn, that the factor imposing difficulties on the GMM representation is the particular sampling characteristics present in this set's point clouds. Because they are composed by three overlapping scans, there are point regions which are denser than others. For instance, relatively dense regions on walls are sometimes seen as linear, instead of planar, due to the concentration of points along a linear direction, rather than distributed uniformly along a plane.

B. Caylus Results

For the Caylus dataset, the grouping resulted in six final classes: *road*, *grass*, *trunk-people*, *vegetation*, *wall* and *far-vegetation*. The results can be found in figure 3. In terms of the relevance of classes, the results do not match those of Freiburg, but are still interesting and consistent with the expectations. The class *trunk-people* represents objects of different natures, such as tree trunks and the human operator, but also building corners. This is a consequence of their similar appearance under the features employed.

In regions far from the sensor, classes become less distinguishable. Tree foliage can appear as *vegetation* if close, or as *far-vegetation* if, well, far. Indeed, the class *far-vegetation* corresponds to a merge of tree canopy and grass. Walls and road, when distant, are also mixed, whereas they are well distinguished when close. The main reason behind these difficulties is the sampling sparsity characteristic of the Velodyne lidar, which increases with the distance. Clutter also had a negative impact, especially on the detection of trunks because these are frequently surrounded by other objects.

C. Discussion

The discovered classes in all of the test cases do correspond to relevant elements in the scene, because they correspond to meaningful geometrical patterns and could thus be used for a given target task. This confirms the

	unknown	ground	wall-building	pole-trunk-people	foliage-vegetation-bicycle	recall	$\mathbf{F_1}$
ground	5959	389651	39	68	1059	0.97	**0.98**
wall-building	30190	689	130764	393	3883	0.78	**0.85**
pole-trunk-people	5179	149	3655	4983	1819	0.31	**0.46**
foliage-vegetation-bicycle	67359	5889	6494	275	71124	0.46	**0.61**
precision	-	0.98	0.93	0.87	0.91		**Total $\mathbf{F_1}$ = 0.72**

Freiburg results with the single-scale model. Accuracy = 0.82.

	unknown	ground	wall-building	pole-trunk-people	foliage-vegetation-bicycle	recall	$\mathbf{F_1}$
ground	15768	381861	108	21	1938	0.95	**0.97**
wall-building	25507	449	139024	164	2241	0.83	**0.89**
pole-trunk-people	5605	116	2029	5805	2328	0.36	**0.53**
foliage-vegetation-bicycle	73706	1051	2693	204	76893	0.50	**0.64**
precision	-	1.00	0.97	0.94	0.92		**Total $\mathbf{F_1}$ = 0.76**

Freiburg results with the multi-scale model. Accuracy = 0.82.

TABLE I

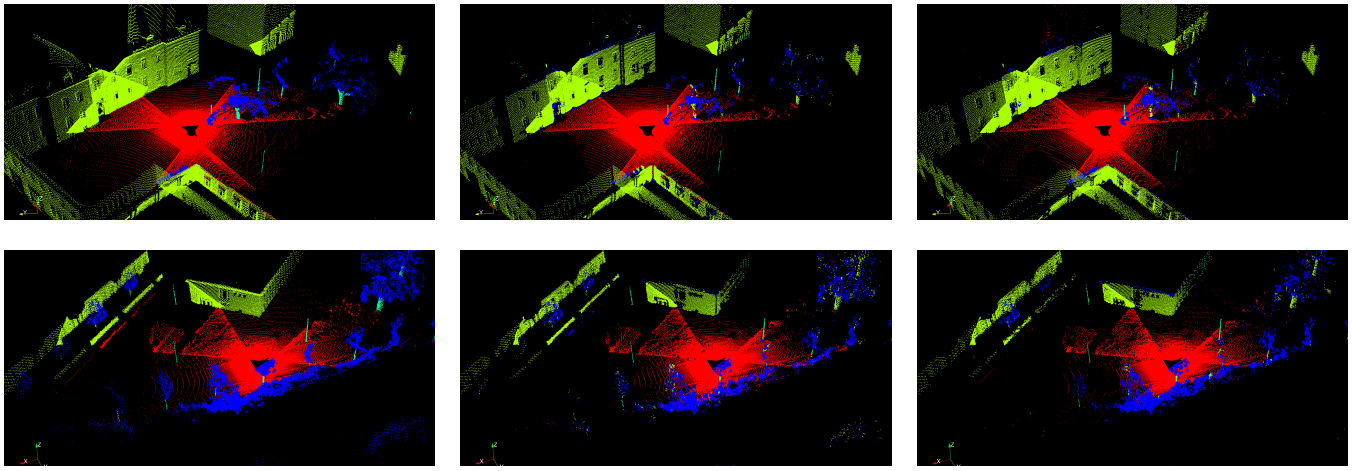

Fig. 2. Freiburg results. Two scans from the validation set. From left to right: ground-truth, prediction of the single-scale model, and prediction of the multi-scale model. Colours: (red, ground), (green, wall-building), (cyan, pole-trunk-people), (blue, foliage-vegetation-bicycle).

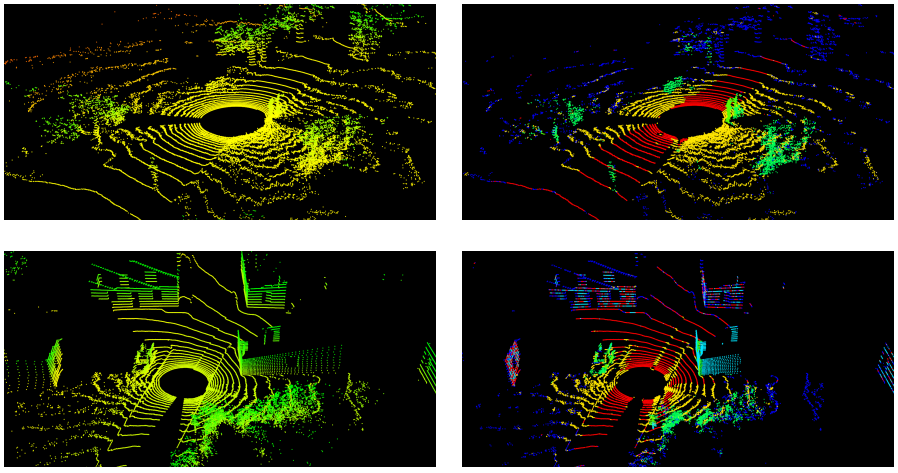

Fig. 3. Caylus results. Two scans from the validation set. Left: original scans, points coloured by height. Right: prediction, colours (red, road), (yellow, grass), (green, trunk-people), (cyan green, vegetation), (light blue, wall), (dark blue, far-vegetation).

capacity of an unsupervised GMM to reveal the natural classes present in the data, and the possibility of finding, in the grouping process, a semantic interpretation for those classes.

In both datasets, the performance of the classification decreases with the distance. This is a consequence of the "myopic" characteristics of lidars, from both the sampling and the precision points of view. The sampling density, which decreases according to the distance, certainly influences the classification in a strong way.

It should be noted that these limitations can be tackled by increasing the number of components of the GMM. Using fewer intermediary classes leads to situations of under-partitioning, where one class corresponds to two or more unrelated objects, and thus cannot be semantically interpreted. Using more classes allows the system to model more precise patterns in the data, which may lead to the distinction between two classes which were previously indistinguishable. The trade-off limiting this flexibility is having a model which requires more data to train, is slower to train, and is slower in inference time.

VI. CONCLUSION

We have proposed a two-layer classification model for outdoor 3D lidar data. The intermediary layer is composed by a GMM trained in an unsupervised manner, and is thus data-oriented. The final layer groups the intermediary classes into higher-level classes which are useful in the target task, and is trained by a human expert, being thus task-oriented. The system is thus naturally flexible, and does not require manual labelling of data.

There are numerous directions in which we can improve the performance of the approach. One direction is to enrich the feature space. Additional features can be easily integrated and will undoubtedly enhance the overall classification process. For the Caylus dataset, acquired with the Velodyne sensor, an interesting feature could be the local shape of consecutive points acquired by a single laser beam, a direction already explored in [17].

Our system performs a pointwise classification, and as such, does not require any pre-processing or post-processing stage. However, adding a post-processing stage might improve the results. A possible post-processing stage could be a filtering based on a voting scheme, as done in [7]. Another possibility could be the use of higher-level information about the class patterns for correcting some misclassifications, such as the classification of corners, windows and doors of buildings as linear objects.

Lastly, the concept of the two-layer model follows in a way the approach taken in coding-based learning methods [22]. These methods also use an intermediary model which represents the data through basic components, called codes or words. The set of all basic components is called codebook or vocabulary. In our case, the basic components would be the Gaussian components of the GMM. It would thus be interesting to examine if our system could be improved by adding to it elements from such approaches, or if using an unsupervised GMM to generate the dictionary could benefit these methods.

REFERENCES

[1] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, 2013.

[2] M. Himmelsbach, T. Luettel, and H. Wuensche, "Real-time object classification in 3d point clouds using point feature histograms," in *IROS*, 2009.

[3] B. Steder, M. Ruhnke, S. Grzonka, and W. Burgard, "Place recognition in 3d scans using a combination of bag of words and point feature based relative pose estimation," in *IROS*, 2011.

[4] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 11, 2008.

[5] J. Behley, V. Steinhage, and A. Cremers, "Performance of histogram descriptors for the classification of 3d laser range data in urban environments," in *ICRA*, 2012.

[6] F. Moosmann and M. Sauerland, "Unsupervised discovery of object classes in 3d outdoor scenarios," in *ICCV Workshops*, 2011.

[7] J.-F. Lalonde, N. Vandapel, D. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *Journal of Field Robotics*, vol. 23, no. 10, 2006.

[8] E. Lim and D. Suter, "3d terrestrial lidar classifications with super-voxels and multi-scale conditional random fields," *Computer-Aided Design*, vol. 41, no. 10, 2009.

[9] N. Brodu and D. Lague, "3d terrestrial lidar data classification of complex natural scenes using a multi-scale dimensionality criterion: Applications in geomorphology," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 68, no. 0, 2012.

[10] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion," in *IEEE Intelligent Vehicles Symposium*, 2009.

[11] A. Aijazi, P. Checchin, and L. Trassoudaine, "Segmentation based classification of 3d urban point clouds: A super-voxel based approach with evaluation," *Remote Sensing*, vol. 5, no. 4, 2013.

[12] D. Munoz, N. Vandapel, and M. Hebert, "Onboard contextual classification of 3-d point clouds with learned high-order markov random fields," in *ICRA*, 2009.

[13] R. Triebel, R. Paul, D. Rus, and P. Newman, "Parsing outdoor scenes from streamed 3d laser data using online clustering and incremental belief updates," in *AAAI Conference on Artificial Intelligence*. AAAI, 2012, pp. 2088–2095.

[14] M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard, "Unsupervised learning of compact 3d models based on the detection of recurrent structures," in *IROS*, 2010.

[15] C. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006.

[16] R. Unnikrishnan, "Statistical approaches to multi-scale point cloud processing," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, 2008.

[17] F. Neuhaus, D. Dillenberger, J. Pellenz, and D. Paulus, "Terrain drivability analysis in 3d laser range data for autonomous robot navigation in unstructured environments," in *ETFA*, 2009.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] K. Moreland, "The paraview tutorial," Sandia National Laboratories, Tech. Rep. SAND 2013-6883P, 2013.

[20] R. Rusu and S. Cousins, "3d is here: Point cloud library (PCL)," in *ICRA*, 2011.

[21] K. Klasing, D. Althoff, D. Wollherr, and M. Buss, "Comparison of surface normal estimation methods for range sensing applications," in *ICRA*, 2009.

[22] A. Coates and A. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *ICML*, 2011.