MMM-classification of 3D Range Data

Anuraag Agrawal, Atsushi Nakazawa, and Haruo Takemura

Abstract— **This paper presents a method for accurately segmenting and classifying 3D range data into particular object classes. Object classification of input images is necessary for applications including robot navigation and automation, in particular with respect to path planning. To achieve robust object classification, we propose the idea of an object feature which represents a distribution of neighboring points around a target point. In addition, rather than processing raw points, we reconstruct polygons from the point data, introducing connectivity to the points. With these ideas, we can refine the Markov Random Field (MRF) calculation with more relevant information with regards to determining "related points". The algorithm was tested against five outdoor scenes and provided accurate classification even in the presence of many classes of interest.**

I. INTRODUCTION

Work is ongoing with regards to machine scene recognition in the field of computer vision and robotics as it is important for a robot to "know" its surroundings for such essential tasks as navigation and manipulation. For example, a robot in an open area can use object classification results to divise a path to automatically navigate the area. In addition, in search-and-recover tasks, identification of the target object is essential to successfully finding the target. A fundamental requirement for achieving such knowledge is to be able to segregate and classify objects within a scene. Work in cognitive science suggests that humans use specific features, including shape, color, and pattern to differentiate between objects[1]. The computer vision community has adopted this into the standard classification method of extracting features from an input method for use in a learning-based approach. However, much of the previous work has focused on classification of objects in 2D images taken from normal cameras, while there has been comparatively little work done in the classification of 3D range data, with methods being presented only relatively recently[2][3]. 2D images are easily made real-time and offer excellent information about color and pattern, but to take advantage of shape data in object recognition, we feel it is important to develop highly-accurate recognition techniques for robust-to-shape 3D range data. This can be modeled as the problem of assigning a label to a 3D point member of a point cloud.

Extracting useful features from range data first requires the definition of a robust descriptor of *shape*. A common technique in the analysis of 3D points is to bin the space around a certain point and count how many other points fall in each bin[4][5]. Because range images only capture surface shape context, such *shape histograms* can accurately describe the local shape around a given point. We call this point-based classification by local descriptors *micro-classification* as it only takes into consideration local data at each point.

Unfortunately, an approach using only local descriptors often generates noisy results where, for example, one point has label a while every other nearby point has label b. Markov random fields have been shown to be a useful tool for solving this problem by taking into account the labels of nearby points when considering a given one, and they have been applied successfully to the classification of range data [6][7]. The Markov random fields in these approaches are set up by connecting points in a nearby radius around a point at random. We call this use of nearby information for use in a local optimization during classification to be *meso-classification*.

In this paper, we present an approach to refine the Markov random field technique for non-local optimization. Previous work in 2D images has shown that taking advantage of shape prior information can improve recognition accuracy by assigning edge potentials based on prepared templates[8]. Similarly in the context of range data, it is appropriate to consider not just the probability a given point belongs to a certain class, but the probability the object that point represents belongs to a certain class. We accomplish this by training not just point features, but object features as well for input range images. We can use these to determine the expected distribution of points relative to the analyzed point in a more global way encompassing the entire object. We call this *macro-classification* and the combination of these three approaches MMM-classification. MMM-classification is able to reduce the appearance of noise patches that a naive Markov random field calculation has difficulty with. A flowchart of the general recognition algorithm is shown in Fig. 1.

Sec. II provides a brief overview of the data acquisition and raw data processing methods used. Sec. III describes the shape histogram local descriptor used to extract local features from the input and Sec. IV explains the calculation of object features for input points. Sec. V outlines the data compression technique we used to maintain a reasonable runtime. Sec. VI shows how to set up the Markov random field for non-local optimization of the result. Sec. VII gives results of the recognition algorithm with respect to certain input scenes, and Sec. VIII concludes the paper and provides some future direction for this work. As an aside, Sec. IX notes the

This work was supported by the Ministry of Internal Affairs and Communications SCOPE project.

A. Nakazawa and H. Takemura are with Cybermedia Center, Osaka University, 1-32 Machikaneyama, Toyonaka, Osaka, Japan {nakazawa, takemura}@cmc.osaka-u.ac.jp

A. Agrawal is with the Graduate School of Information Science and Technology, Osaka University, 1-32 Machikaneyama, Toyonaka, Osaka, Japan anuraag@lab.ime.cmc.osaka-u.ac.jp

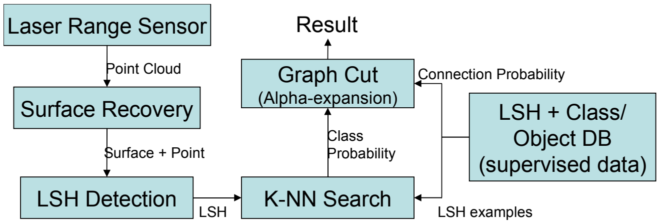


Fig. 1: Flowchart of recognition algorithm

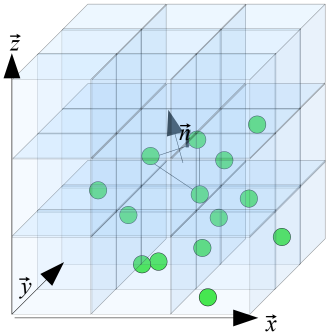


Fig. 2: Visualization of partitioned space around a polygon. Points in each bin are counted and stored in the elements of the histogram feature vector.

utility of GPU acceleration in current vision work.

II. RANGE DATA

Input scenes are captured as laser range data taken by a SICK laser rotated 360^o. The laser rotates up and down along scan lines and returns the distance (range) to any surface intersecting this scan line. As the laser is rotated, scan lines are taken at increasing rotation angles until the range data for the entire surroundings is captured. The laser rotation angle ϕ, scan line angle from the horizontal θ, and range r can be converted into Cartesian coordinates (x,y,z) using a standard polar-Cartesian conversion. While this input image is made up entirely of points (leading to the name point-cloud), it is possible to roughly reconstruct surface polygons from the points by connecting nearby points in adjacent scan-lines as in Fig. 4b. The result of this is an input scene with points $p_i \in P$ and polygons $x_i \in X$. The use of polygons provides the advantage over raw points that there is connectivity information that can be applied to edge potentials in the Markov random field.

III. LOCAL SHAPE HISTOGRAM

The principal local feature extracted from the input is a *local shape histogram*. This is similar to previous approaches that partition space around a target point and count neighboring points falling inside these bins[6]. However, previous approaches orient the partitioned cube of space with respect to the principal plane around the target point. When classifying outdoor scenes, rotation with respect to the vertical usually contains a significant amount of information that can increase recognition accuracy compared to rotation invariant features. Thus, our approach includes the vertical vector when orienting the partitioned cube around a point.

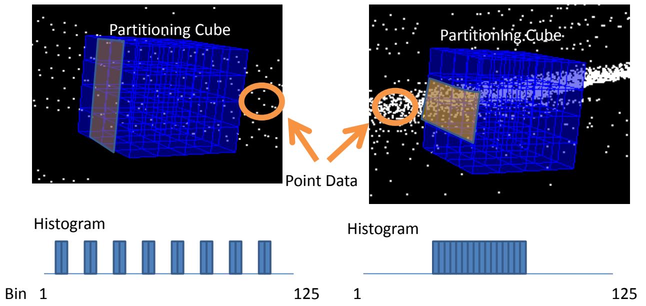


Fig. 3: Conceptual examples of histograms obtained from points. Left is a vertical plane and right is a horizontal plane. White dots represent points in the input range data. The blue cube is partitioning space into $5\times5\times5$ bins. Bins are indexed horizontally, so the left vertical plane results in the below histogram with alternating bins of high frequency. The right horizontal plane, on the other hand, has a histogram with consequetive bins of high frequency.

For every surface in our input set, we take the center point $\vec{t}$, the normal vector $\vec{n}$, and the up vector $\vec{u}$ and define a local coordinate system as in

$$\begin{bmatrix}\vec{x} & \vec{y} & \vec{z} & \vec{t}\end{bmatrix} = \begin{bmatrix}(\vec{n}\times\vec{u})\times\vec{u} & \vec{n}\times\vec{u} & \vec{u} & \vec{t}\end{bmatrix} \tag{1}$$

These local coordinates form the basis vectors for the partitioning space as shown in Fig. 2. Points contained in each bin of the cube are counted and stored as a normalized multi-dimensional histogram (i.e. if the dimensions of the partitioning cube is $d\times d\times d$, then the histogram is a d^3 dimensional vector). This histogram can express the three-dimensional shape around a given point while being invariant to rotation around the vertical axis. Fig. 3 shows two examples of the histogram obtained from a vertical plane and a horizontal plane respectively. The differing shape results in sharply different histograms.

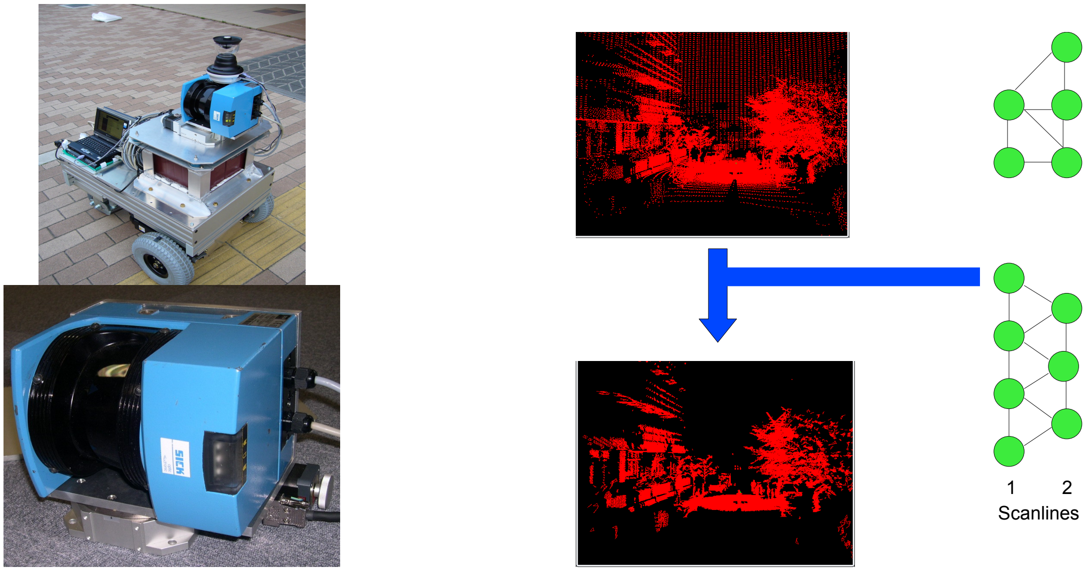
Due to the need for obtaining all points within a defined area (in this case, the partitioning cube), we inserted all of the points into a KD-tree for high-performance distance-based lookups.

IV. OBJECT FEATURES

To train object features for a training data set, we first manually segment the data into separate objects. Then, for each object o_i, we compute the covariance matrix of the distribution of points composing it as in

$$\mathbf{COV_i} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})(x_i-\bar{x})^T \tag{2}$$

where x_i is the 3D Cartesian coordinate for each point in the object. The rows of the covariance matrix can be used to define a 3D box containing all of the points, providing a rough expression of the shape of the object. The label of the



(a) Robot

(b) Surface reconstruction

Fig. 4: Left is a picture of the data acquisition robot and range sensor. Right shows surface reconstruction from the point cloud. Nearby points in adjacent scan-lines are connected as triangles.

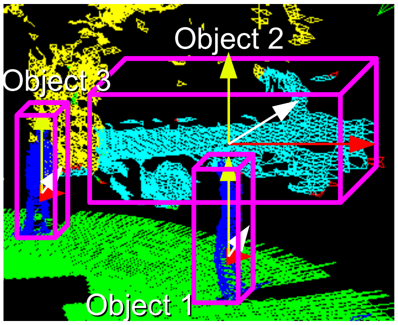


Fig. 5: Segmentation of objects in a scene. Coordinate axes indicate basis vectors obtained from the covariance matrix.

object is the label l of all the points comprising it. As such, the object feature for o_i is simply $[\mathbf{COV_i} \quad l]$. Fig. 5 shows a sample manual object segmentation and boxes obtained from the basis vectors of the covariance matrix.

During recognition, input range images cannot be manually segmented into objects. As such, it is necessary to use an auto-segmentation technique to find objects in the input scene. As the range data has already been triangulated into polygons, as described in II, it is trivial to search along neighboring polygons to find all connected polygons x_j that make up an object o_i and calculate $\mathbf{COV_i^{in}}$. Then, we match $\mathbf{COV_i^{in}}$ against all of the features in the object database using a Bhattacarrya distance function. A histogram of the k nearest entries is used to construct object class distribution $P_i^o(l)$ by counting up the number of near neighbors for each class and normalizing to form a probability distribution.

V. LOCAL FEATURES

While it is important to create an extensive database encompassing a large sample of data to ensure high-accuracy recognition, the processing time of the algorithm is directly related to the size of this database. In order to take into account a large sample of data while maintaining a reasonable execution time, we employ a vector quantization technique to compress the sample features into representative clusters using the k-means++ algorithm to reduce the appearance of empty clusters[9]. Using a codebook has been shown to provide accurate results with relatively low execution time[10].

We ran clustering with respect to the local shape histograms (LSH) for each polygon. A histogram represents the shape feature of the polygon, so similar polygons will have similar histograms and should end up in the same cluster after k-means processing. After generating k clusters, it is important to determine the features associated with each cluster. Averaging the histograms of each element in the cluster can produce a reasonable representative shape feature for the cluster.

To prevent one class from dominating the others by having a greater number of points among the training scenes, we ran the clustering on a class-by-class basis. For each class label, l, we produce k clusters of the histograms of all polygons x_i with label l. The shape feature of the cluster becomes $\bar{h}$, the average of all of the histograms of the elements in the cluster. The label for the class is the label of the elements in the cluster. As such, the resulting database has kL rows of $[\bar{h} \quad l]$, where L is the total number of classes.

Matching codebook entries are found for each input histogram using a brute force k-nearest-neighbors search for each input histogram as shown in Fig. 6. More sophisticated methods such as KD-trees are not appropriate due to the high dimensionality of the feature vector. n nearest codebook entries are found for each input histogram, and the number of matching entries for each class are counted up

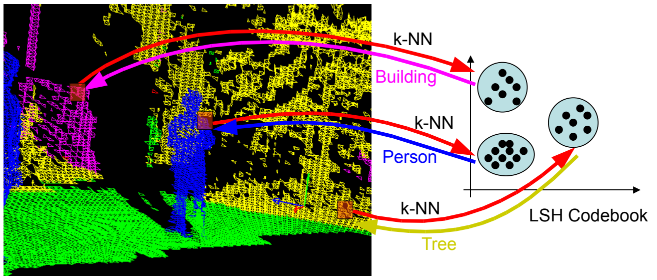


Fig. 6: A local shape histogram codebook. Patches in the histogram are matched to clusters in the codebook, and the patch is then labeled with the cluster's class. In this case, the cluster's class is the member of its class probability distribution with the highest probability.

and normalized to produce class distribution $P_i^s(l)$ for input histogram h_i.

VI. MARKOV RANDOM FIELD

$P_i^s(l)$ and $P_i^o(l)$ are used to set up the potentials of a Markov network for final optimization. Markov networks can be modeled as in Equation 3.

$$P(l) = \frac{1}{Z} \prod_{i=1}^{N} \phi_i(l_i) \prod_{ij \in \varepsilon} \phi_{i,j}(l_i, l_j) \tag{3}$$

Here, $\phi_i(l_i)$ is the tendency of node i to take on the label l_i and $\phi_{i,j}(l_i, l_j)$ is the tendency of two nodes with labels l_i and l_j respectively to be connected in the network. $\phi_i(l_i)$ can be naturally expressed as a function $p_i(l)$, the probability node i is an instance of class l. This notion of probability is what led us to create probability distributions for each node in the previous sections rather than using a discriminative method similar to what has been employed in previous work[6].

Each polygon in the input scene is represented as a node in the markov network with node potential as defined below:

$$\phi_i(l_i) = w_s P_i^s(l_i) + w_o P_i^o(l_i) \tag{4}$$

w_s and w_o are weights given to the shape class distribution and object class distribution respectively. Edge potentials are defined as

$$\phi_{i,j}(l_i, l_j) = \begin{cases} \phi_L & \text{if } l_i = l_j \\ \phi_S & \text{if } l_i \neq l_j \end{cases} \tag{5}$$

where ϕ_L and ϕ_S are user-defined constants such that $\phi_L \geq \phi_s$.

We solve for a pseudo-optimal configuration for the Markov network using the alpha-expansion procedure introduced by Boykov et al.[11] that uses an iterative minimum-cut algorithm to guarantee a factor 2 approximation of the optimal solution. A visualization of an alpha expansion pass is shown in Figure 7.

VII. EXPERIMENT

To test the effectiveness of the proposed method, we took nine outdoor scenes of Osaka University with a rotating SICK LMS-200 laser range finder mounted on a mobile robot. We selected these scenes in such a way that we could obtain a variety of samples for each of the classes under consideration. We then manually labeled all the points in all of the scenes with the correct class. In addition, we identified objects in the scene and assigned a unique label for each of these. We then selected representative data from four of these scenes so that we ended up with a database of 237177 points with their respective histograms and object features. Histograms were calculated by a $80cm \times 80cm \times 80cm$ block partitioned into $8 \times 8 \times 8$ bins. The histogram database was then clustered into 100 clusters for each class resulting in a final training database of 1K representative centroids. Probability distributions for input nodes were calculated from the 100 nearest matching codebook entries. The training images contained 89 objects whose features were placed in the object database as is. Input object features were matched against their one closest database entry as the object database was still relatively small. ϕ_s, ϕ_L, w_s, and w_o were 0.1, 1.5, 1.0, 0.2 respectively.

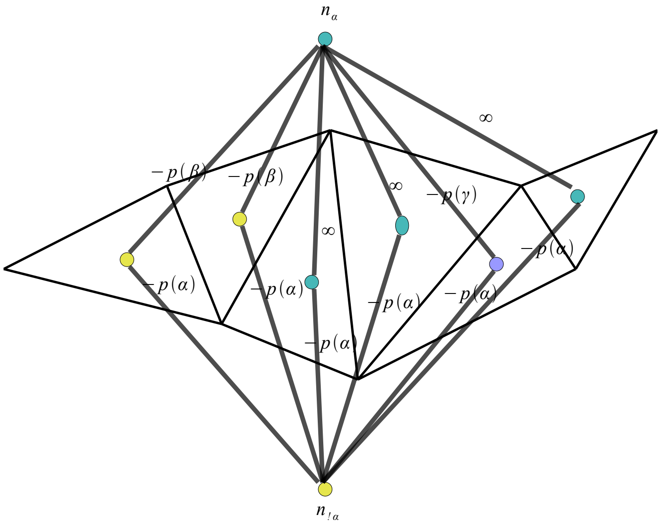


Fig. 7: Example alpha expansion graph. Teal nodes are class α, yellow nodes are class β, and the violet node is class γ.

We then inputted the five remaining scenes into our recognition algorithm to infer the class labels for every point. These inferred labels were compared with manually assigned ground truth labels. Recognition accuracy for each class within each scene is shown in Table I for MMM-classification and in Table II for classification without object features using only local shape histograms and alpha expansion. A color-coded visualization of the result is shown in Figure 8.

Recall rates remained more or less consistent between the proposed and previous method. However, the recall rate for car was much improved by using object features. This was expected, as previous misrecognition of car usually involved doors being detected as building walls, but the object feature for car is able to match the entire vehicle, eliminating the building misrecognition in many cases. Step has a low recall rate, oftentimes misrecognized as car, but there are not enough samples in the test set to make a clear judgment

TABLE I: Total recognition rate for MMM-classification

		Detected						Recall
		Ground	Tree	Building	Step	Car	Person	
Actual	Ground	284971	144	104	10	1	36	99.90%
	Tree	147	137121	2607	5	105	384	97.69%
	Building	299	10010	72185	0	88	269	87.12%
	Step	5	59	2	203	133	0	50.50%
	Car	50	32	1398	0	14628	241	89.47%
	Person	15	249	0	0	0	26684	99.00%
Precision		99.80%	92.88%	94.61%	91.44%	97.81%	94.30%	

TABLE II: Total recognition rate for only local shape histograms with alpha expansion

		Detected						Recall
		Ground	Tree	Building	Step	Car	Person	
Actual	Ground	284971	144	104	10	1	36	99.90%
	Tree	148	132492	6913	5	356	455	94.39%
	Building	299	9835	72288	0	79	299	87.33%
	Step	5	59	2	203	129	4	50.50%
	Car	50	478	2466	1	13070	284	79.95%
	Person	15	249	0	0	0	26684	99.02%
Precision		99.80%	92.46%	88.40%	91.03%	95.86%	93.97%	

as to why.

VIII. Conclusion and Future Work

A. Conclusion

In this paper we have proposed an approach for the classification of range images into object types using a Markov random field with shape priors. The proposed MMM-classification method first performs micro-classification based on local shape descriptors in the form of shape histograms. Then, it performs meso and macro-classification by connecting neighboring points and more distant related points in a Markov network and optimizes the labeling by solving the maximum a-posteriori inference problem via alpha-expansion and graph-cuts. The proposed method can accurately label complicated outdoor scenes even when taking into account a relatively high number of classes. In particular, it is effective at distinguishing cars and buildings, two classes with similar local features but drastically different object ones.

B. Future Work

One major improvement that can be made is to add a global classification step that attempts to assign probabilities to configurations of semantic arrangements of objects. For example, a tree trunk will usually be under a tree canopy, so a high probability can be assigned to this configuration. Likewise, it would be very unlikely for a person to be under a tree canopy, so a low probability can be assigned to such a configuration. These global preferences could reduce some of the more egregious mis-recogntions that showed up in the experiment.

Another possible future direction is to add sub-classes that are not targets of interest but could be used during recognition. For example, adding subclasses such as tires and vertical wall to car and building respectively could open up possibilities to define potentials that dissuade impossible configurations such as when a car's tires are resting on a vertical wall. Adding classes always increases the difficulty of recognition, but if these additional classes were confined to a separate recognition step like the above example, their negative impact on recognition would likely be minimal.

We would also like to examine other choices for the object feature. Covariance matrices are incredibly simple, and while they were enough to show the utility of an object feature in range data classification, more sophisticated object descriptors could produce better results.

Finally, it would be interesting to integrate this classification algorithm into a robot navigation system to see how well the object recognition can be used in a practical application. In particular, practical testing could identify new classes of interest to be considered in future experiments and algorithmic improvements.

IX. GPU Acceleration

As an aside, we would also like to mention the benefits of GPU acceleration in computer vision work. The k-means++ clustering and k-nearest-neighbors matching algorithms were implemented as GPGPU kernels executable on nVidia 8-series and above video cards. The extremely parallel nature of computer vision algorithms are highly suited to execution on video cards. We only did informal testing, but k-means++ on a data set of size 200K being clustered into 1000 clusters produced a roughly 6x speedup, taking approximately six hours on the CPU and less than one hour on the GPU. k-nearest-neighbors between a codebook of size 200K and an input image of size 200K also produced similar speedups, going from approximately six hours to one hour. We would like to do more formal performance analysis in the future in addition to more advanced optimization of the GPU kernels, but it is apparent that the GPU is a powerfool tool to take advantage of in modern algorithms that can either simply increase the rate at which experiments can be completed or

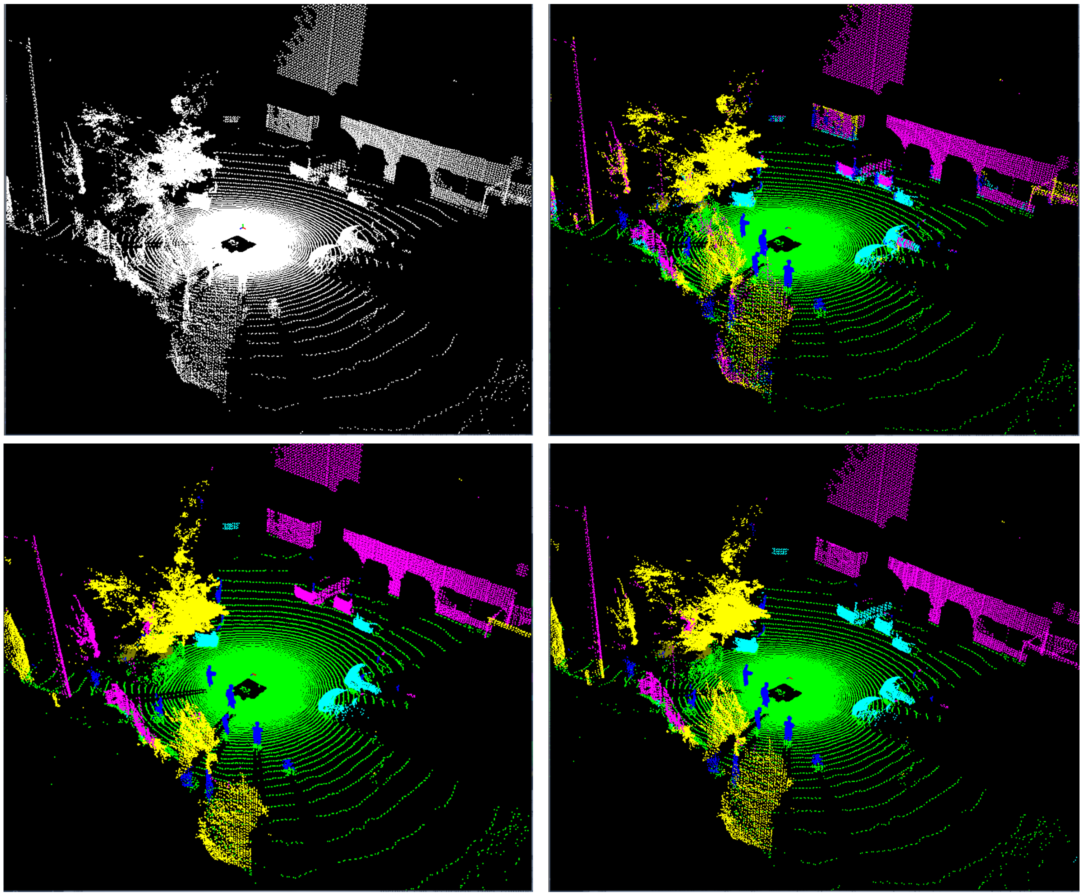


Fig. 8: Color-coded recognition result. Top-left is input range image. Top-right is recognition with only local features. Bottom-left is recognition with local features and alpha expansion. Bottom-right is recognition with local features, object features, and alpha expansion (MMM-classification).

even make traditionally slow algorithms executable in real-time.

X. ACKNOWLEDGMENTS

This work was supported in part by the Strategic Information and Communications R&D Promotion Programme (SCOPE), Japan.

The authors gratefully acknowledge the contribution of the Ministry of Internal Affairs and Communications.

REFERENCES

[1] T. Wilcox, "Object individuation: infants' use of shape, size, pattern, and color," *Cognition*, vol. 72, pp. 125–166, Sept. 1999. [Online]. Available: http://www.sciencedirect.com/science/article/B6T24-3XM2T3P-2/2/e3be57a2261484960ca649310b7588dc

[2] N. Vandapel, D. Huber, A. Kapuria, and M. Hebert, "Natural terrain classification using 3-d ladar data," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 5, 2004, pp. 5117–5122 Vol.5.

[3] D. Munoz, N. Vandapel, and M. Hebert, "Directional associative markov network for 3-d point cloud classification," in *Fourth International Symposium on 3D Data Processing, Visualization and Transmission*, June 2008.

[4] A. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, pp. 433–449, 1999.

[5] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik, *Recognizing objects in range data using regional point descriptors*, 2004. [Online]. Available: http://citeseer.ist.psu.edu/frome04recognizing.html

[6] 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 *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, 2005, pp. 169–176 vol. 2.

[7] R. Triebel, K. Kersting, and W. Burgard, "Robust 3d scan point classification using associative markov networks," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 2603–2608.

[8] N. Vu and B. Manjunath, "Shape prior segmentation of multiple objects with graph cuts," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, 2008, pp. 1–8.

[9] D. Arthur and S. Vassilvitskii, "k-means++: the advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035. [Online]. Available: http://portal.acm.org/citation.cfm?id=1283383.1283494

[10] J. Liebelt, C. Schmid, and K. Schertler, "Viewpoint-independent object class detection using 3d feature maps," in *IEEE Conference on Computer Vision & Pattern Recognition*, 2008. [Online]. Available: http://lear.inrialpes.fr/pubs/2008/LSS08

[11] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, pp. 1222–1239, 2001.