Plane-based Object Categorization using Relational Learning

Reza Farid, Claude Sammut

School of Computer Science and Engineering
The University of New South Wales
Sydney, NSW 2052 Australia

{rezaf,claude}@cse.unsw.edu.au

Abstract- We use Inductive Logic Programming (ILP) to learn classifiers for generic object recognition from point clouds, as generated by 3D cameras, such as the Kinect. Each point cloud is segmented into planar surfaces. Each subset of planes that represents an object is labelled and predicates describing those planes and their relationships are used for learning. Our claim is that a relational description for classes of 3D objects can be built for robust object categorization in real robotic application. To test the hypothesis, labelled sets of planes from 3D point clouds gathered during the RoboCup Rescue Robot competition are used as positive and negative examples for an ILP system. The robustness of the result is evaluated by 10-fold cross validation. The results show that ILP can be successfully applied to recognise objects encountered by a robot in an urban search and rescue environment.

Keywords- object classification, inductive logic programming, ALEPH, cross validation, urban search and rescue, point cloud, machine learning, range data.

1 Introduction

The goal of this work is to use machine learning to build an object classifier for an autonomous robot in an urban search and rescue operation. The primitive input to the classifier is a 3D range image, representing a partial view of the environment. Generic object recognition requires representations of classes of objects and a classification method to recognise new objects. Object features may be visual, structural, functional, etc.

3D depth cameras, such as the Microsoft Xbox Kinect, are now becoming widely used because they provide both range and video images and their cost is much reduced compared with previous generations of such cameras. In a range image, each pixel's value represents the distance of the sensor to the surface of an object in a scene from a specific viewpoint [1, 2]. This can be used to infer the shape of the object [3]. The Kinect, also incorporates a colour video camera but in this paper, we only use the depth information for object recognition as colour calibration under different lighting conditions is problematic [4]. A range image can be transformed into a set of 3D co-

ordinates for each pixel, producing a point cloud. Fig. 1 shows a range image of a staircase with four steps, taken by a robot positioned in front of the staircase. The figure includes front and top views for the same point cloud. The point cloud has been segmented into planes that are identified by unique colours. A range image only provides a partial view of a scene, since it is taken from one viewpoint. Constructing a complete 3D point cloud for an object requires multiple views.

In our current experiments, we extract planes from the 3D point cloud and use them as primitives for object recognition. Planes are useful in built environments, including urban search and rescue for identifying floors, walls, staircases, ramps and other terrain that the robot is likely to encounter. An ILP system is then used to discover the properties and relationships between the planes that form an object. In the following sections, we describe the plane extraction method, the learning algorithm and the experimental results that demonstrate the utility of this approach.

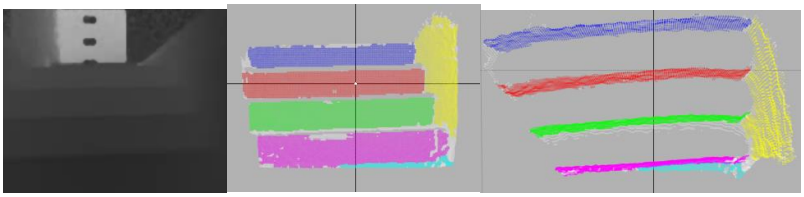

Fig. 1. Range image and its correspondent point cloud (coloured) from front and top view

2 Background and Related Work

A considerable amount of research has been devoted to generic object recognition [4-7], which is required by robots in many tasks. For example in service robotics applications, such as a catering or a domestic robot [6], the robot must recognise specific kinds of tableware. In industrial applications, the robot has to distinguish a set of products [7]. We are mostly interested in urban search and rescue (USAR); where a team of robots are sent to a post-disaster environment. The robots are expected to search autonomously and provide valuable information especially to the human rescuers. The RoboCup Rescue Robot League [8] is a competition which provides the environment for testing and motivating the related research in USAR [9].

In recent years, statistical methods such as SIFT [10] have become popular methods for object recognition. However, these are limited to recognising individual objects that have been previously seen and stored in the vision system's database. In generic object recognition (GOR) [11], the system learns to recognise an object as belonging to a generic class [12], as one would expect in concept learning. Relational learning is well suited to learning object classes, provided that the primitive features needed for recognition can be reliably extracted from the image.

Our approach is most closely related to Shanahan [13, 14] who uses a logic program as a relational representation for 3D objects in 2D line drawings, and abduction is used in object recognition. We have extended this representation, replacing the 2D lines with 3D planes. Furthermore, we use ALEPH [15] to learn the logic programs

from instances obtained by a robot equipped with a depth camera. Originally this was a SwissRanger SR-3000 camera [16], which has now been replaced by a Kinect. The robot is shown in Fig. 2. It was designed to participate in the RoboCup Rescue Robot competition [8], held annually. The competition area uses elements designed by the US National Institute of Standards and Technology (NIST) [17] to certify robots for emergency operation. These elements are typical of hazards that might be expected in buildings damaged by a disaster such as an earthquake.

Robots developed for the rescue competition have been successfully deployed in the Fukushima nuclear reactor damaged in the earthquake and tsunami that struck Japan in March 2011. A portion of a typical arena is shown in Fig. 3. The task for the robot is to traverse the arena, searching for victims while making a map of the area. Rescue robots may be tele-operated or autonomous. Our rescue robot team has won the RoboCup award for best autonomous robot three years in succession 2009-2011.

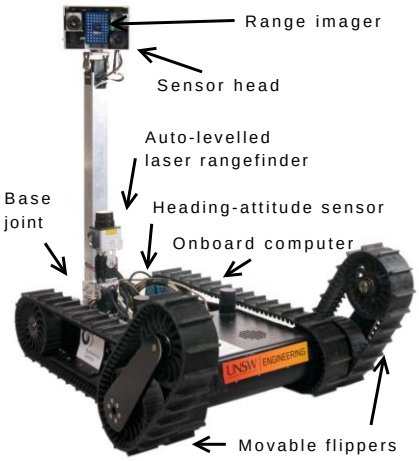

Fig. 2. The rescue robot platform

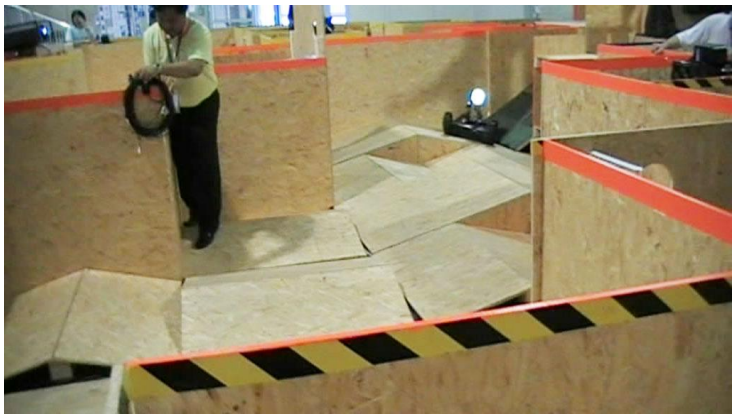

Fig. 3. Roll and pitch ramps in a RoboCup rescue arena

When running autonomously, recognition of objects is essential so that the robot knows how to behave. For example, recognising a staircase tells a wheeled robot that it must avoid that object, whereas a tracked robot, such as the one in Fig. 2 is capable of climbing stairs but it must reconfigure the flippers to be able to climb successfully. A relational representation is useful in this application because we wish to recognise objects that are characterized by relationships between its parts, as in the steps that constitute a staircase, and the number of parts may not be fixed, as the number of steps in a staircase can vary.

Before discussing the ILP methods used to learn to recognise objects in the arena, we first discuss the pre-processing that is essential for the ILP to work.

3 Feature Extraction

We use the plane as the primitive for describing objects, where an object is considered to be composed of a set of planes derived from a point cloud. To find these planes, each point's normal vector is calculated and used to segment the point cloud. That is, neighbouring points are clustered by the similarity of their normal vectors. Fig. 1 shows an example of planes found using this method. Attributes of the planes are then calculated, including the spherical representation of the planes normal vector and the relationships between pairs of planes, e.g. the angle separating them. After extracting these features, sets of plane are labelled according to the class to which they belong. The ALEPH ILP system [15] builds a classifier for each class, where objects belonging to that class are considered positive examples and all other objects are treated as negative examples.

In addition to the spherical representation of a plane's normal vector (θ and φ) [18] (note that θ is defined as zero for undefined situation), other attributes are derived from the convex hull of the plane. These are the diameter and width of the convex hull and the ratio between these values. The plane's bounding cube is used to calculate the ratios between the three axes, two by two. The final plane feature is the axis along which the plane is most distributed. Fig. 4 shows the results of segmentation, convex hull creation and normal vector representation for a scene that contains a pitch/roll ramp and maze wall objects.

After planes are found and their individual attributes are calculated, we then construct relations between each pair of planes. The first relation is the angle between the normal vectors of each pair of planes. The second is a directional relationship [19, 20] that describes how two planes are located with respect to each other. For example, as shown in Fig. 5, rectangle B is located on the 'east' side of rectangle A. To find this direction between two rectangles, the line that connects both centres is used as a *base* to find out which *direction line* is closer to the base. However, in our method, regions (planes) are in 3D space, not 2D space. We project the 3D view onto two 2D views and find spatial-directional relationships in each 2D view. A bounding cube, with respect to the sensor's coordinate frame, is generated for each set of points assigned to a plane. Then, two projections of this cube are used to represent the minimum bounding rectangles for the region from each of the two views. The projections are on the

XY plane (front view) and the ZX plane (top view). Having two 2D views of a 3D object is sufficient to represent its bounding rectangles.

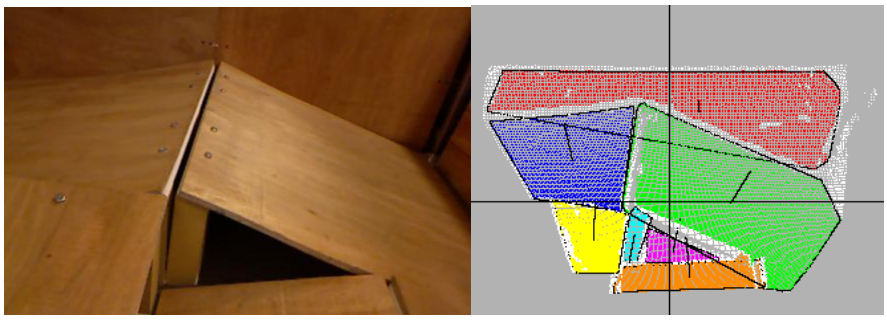

Fig. 4. Colour image and segmented point cloud (front view); representing convex hull and normal vector for each region

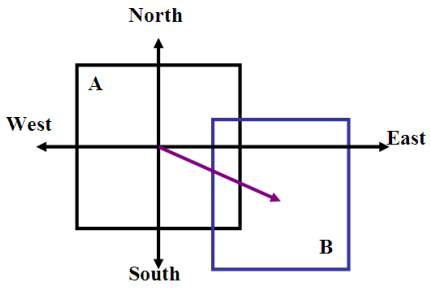

Fig. 5. east(B,A)

4 Learning Object Classes

To label training and test examples, we developed a user-interface that processes each range image, converts it to a point cloud and then shows the result of point cloud segmentation to a human trainer who chooses a set of coloured regions to form a positive example of an object and a negative example for some other objects. The trainer labels set of planes with the class to which the object belongs, e.g. staircase, wall, and pitch/roll ramp. ALEPH is used to construct one classifier for each type of object. For example, to learn to recognise a staircase, all the objects labelled as staircases are treated as positive examples and all other objects are considered negative examples. In these experiments, the range images are obtained from a Microsoft Xbox Kinect. The same method has also been applied to images from a SwissRanger SR-3000.

After labelling, the data set is ready to be used for learning. The labelled planes are represented as a Prolog list called as 'planeset'. Fig. 6 shows the legend for colours used, the point cloud segmentation result as each region's convex hull and normal vector and its correspondent colour image for a Kinect box. The red region, region 1,

represents the wall, while the yellow region, region 4, represents the top of a desk. For this example, we define a positive example for *box* that includes regions 2, 3 and 5 in image, *img_00*, creating the predicate `class(box,img_00,[pl2,pl3,pl5])`. That is, predicates of the form `class(#class,+image,+planeset)` represent an unordered list of planes (or 'planeset') in an image forming an instance of the object class.

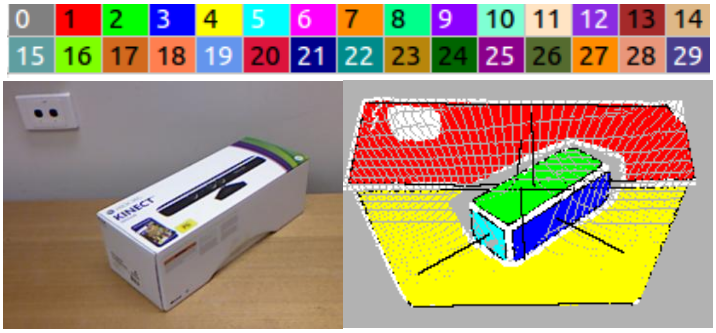

Fig. 6. Colour legend (top), colour image and segmented point cloud (front view) for box

This object is described in the learner's background knowledge by a set of predicates, where the first set of predicates specifies the image's number and planes. Note that, to make each plane identical, we use a pair based on the image number and plane number.

```
image(img_00).
plane(pl1).   plane(pl2).   plane(pl3).
plane(pl4).   plane(pl5).
```

The next set of predicates describes the attributes for each plane. The first attribute is on which axis the plane is distributed most. To calculate that, we find the difference between the maximum and minimum value of a region's point coordinates (Δx, Δy and Δz) and compare them to decide which axis should be chosen.

```
distributed_along(img_00,pl1,axisX).
distributed_along(img_00,pl2,axisX).
distributed_along(img_00,pl3,axisX).
distributed_along(img_00,pl4,axisX).
distributed_along(img_00,pl5,axisY).
```

We also use the ratio between each pair of values (Δx/Δy, Δy/Δz and Δx/Δz) as another set of features. However, instead of using the exact value, we bin them in an interval around a pre-defined value. For example both '1.23' and '1.12' are represented by '1±0.25'. This method is applied for another plane feature, the ratio between the diameter and width of the convex hull. The value of the ratio may be negative. For example, for plane 5 in image_00, since Δx<Δz, then the ratio Δx/Δz is represented as negative value of Δz/Δx.

```
ratio_yz(img_00,pl1,'-1.0±0.25').
ratio_xz(img_00,pl1,'4.5±0.25').
ratio_xy(img_00,pl1,'5.0±0.25').
…
ratio_yz(img_00,pl5,'1.0±0.25').
ratio_xz(img_00,pl5,'-1.5±0.25').
ratio_xy(img_00,pl5,'-1.5±0.25').

ch_ratio(img_00,pl1,'4.0±0.25').
ch_ratio(img_00,pl2,'2.5±0.25').
ch_ratio(img_00,pl3,'3.5±0.25').
ch_ratio(img_00,pl4,'2.0±0.25').
ch_ratio(img_00,pl5,'1.5±0.25').
```

The last plane attribute is the spherical representation of its normal vector. Similar to the ratio representation, we use an interval around an angle. For example, both 91.35 and 87.87 are binned in the interval '90 ±15'.

```
normal_spherical_theta(img_00,pl1,'-90±15').
normal_spherical_phi(img_00,pl1,'135±15').
…
normal_spherical_theta(img_00,pl5,'-135±15').
normal_spherical_phi(img_00,pl5,'112±15').
```

As mentioned earlier, relations are derived from pairs of planes: the angle between the normal vectors of two planes and the directional relationship for two adjacent planes from XY and XZ views. Note that, since we use a projection of each plane on XY and XZ, two planes can appear adjacent in one view and not in the other. For the above example, these features are:

```
angle(img_00,pl1,pl2,'90±15').    angle(img_00,pl1,pl3,'45±15').
angle(img_00,pl1,pl4,'90±15').    angle(img_00,pl1,pl5,'45±15').
angle(img_00,pl2,pl3,'90±15').    angle(img_00,pl2,pl4,'0±15').
angle(img_00,pl2,pl5,'90±15').    angle(img_00,pl3,pl4,'90±15').
angle(img_00,pl3,pl5,'90±15').    angle(img_00,pl4,pl5,'90±15').

dr_xz(img_00,pl1,pl2,connected).    dr_xz(img_00,pl1,pl2,west).
dr_xz(img_00,pl2,pl1,east).         dr_xz(img_00,pl2,pl3,west).
  …
dr_xz(img_00,pl5,pl3,connected).    dr_xz(img_00,pl5,pl3,south).
dr_xz(img_00,pl5,pl4,covers).

dr_xy(img_00,pl1,pl2,connected).    dr_xy(img_00,pl1,pl2,north).
dr_xy(img_00,pl1,pl3,connected).    dr_xy(img_00,pl1,pl3,north).
  …
dr_xy(img_00,pl4,pl5,is_covered).   dr_xy(img_00,pl5,pl2,east).
```

```
dr_xy(img_00,pl5,pl4,covers).
```

Regions 2, 3 and 5, which form a box, are perpendicular to each other and this fact appears in their pair-wise angle relationships.

Consider planes 1 and 2 for example of a directional relationship. From both views, front and top, the regions overlap. , Thus, *connected* appears in both directional relationships. Also from the XY view, region 2 (green) is below region 1 (red) and *plane1* is north of *plane2* from the XY view in this image, giving, `dr_xy(img_00,pl1,pl2,north)`. Similarly, projecting these planes in the XZ view and assuming the X-axis represents north-south and the Z-axis represents east-west, *plane1* is west of *plane2* in the XZ view, given by `dr_xz(img_00,pl1,pl2,west)`.

In some cases, the number of planes that form an object may be different. For example, by using the same colouring legend, in Fig. 7, different sets of planes from image *img2* can form positive examples for staircase, represented by following predicates:

```
class(staircase,img2,[pl06,pl08,pl10,pl11]).
class(staircase,img2,[pl05,pl06,pl08,pl10]).
class(staircase,img2,[pl04,pl05,pl06,pl08]).
class(staircase,img2,[pl01,pl03,pl04,pl05]).
class(staircase,img2,[pl01,pl03,pl04,pl05,pl06,pl08]).
class(staircase,img2,[pl03,pl04,pl05,pl06]).
class(staircase,img2,[pl01,pl03,pl04,pl05,pl06,pl08,pl10]).
class(staircase,img2,[pl04,pl05,pl06,pl08,pl10,pl11]).
class(staircase,img2,[pl01,pl03,pl04,pl05,pl06,pl08,pl11]).
```

5 Evaluation

To evaluate the learning system, we use 10-fold cross-validation. The performance of the learning algorithm is measured by the accuracy, error rate, precision and recall [21] as shown in Table 1.

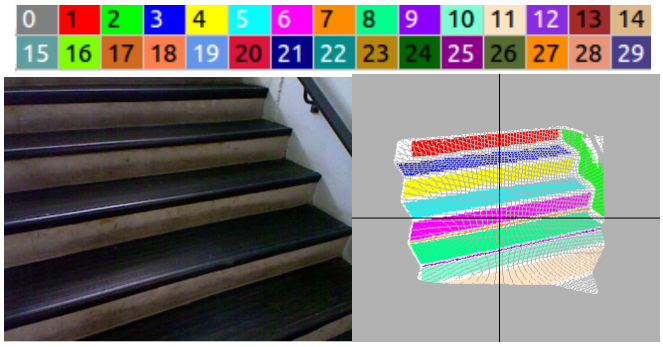

Fig. 7. Stairs with different number of planes

Table 1. Results for 10-fold cross validation

Object	No. positive	No. negative	Accuracy =(TP+TN)/N	Error rate =(FP+FN)/N	Precision =TP/(TP+FP)	Recall =TP/(TP+FN)
Step	199	731	0.9645	0.0355	0.9368	0.8945
Staircase	241	665	0.9956	0.0044	0.9917	0.9917
Wall	105	819	0.9881	0.0119	0.9608	0.9333
Box	144	780	0.9741	0.0259	0.9618	0.8690
Pitch/roll ramp	131	205	0.9464	0.0536	0.9520	0.9084

The classifiers achieve high accuracy because we have ensured that the training data include images taken from several viewpoints. For example, a box, depending on the viewpoint, may appear to have two or three sides. The longest side of the box may be horizontal, vertical or diagonal. By including examples of all these variations, we can train the classifiers to handle different perspectives. The features used in describing objects also affect the generality of the classifier. We construct features as much as possible, which are invariant to transforms and thus, enable the learning algorithm to find general descriptions.

An example of the output from learning to recognise a staircase is:

```
class(staircase, IMAGE_A, PLANESET_B) :-
  member(C, PLANESET_B),          member(D, PLANESET_B),
  angle(IMAGE_A, D, C, '0±15'),   member(E, PLANESET_B),
  angle(IMAGE_A, E, D, '90±15'),  angle(IMAGE_A, E, C, '90±15'),
  distributed_along(IMAGE_A, E, axisX).

class(staircase, IMAGE_A, PLANESET_B) :-
  member(C, PLANESET_B),            member(D, PLANESET_B),
  angle(IMAGE_A, D, C, '0±15'),
  member(E, PLANESET_B),            member(F, PLANESET_B),
  angle(IMAGE_A, F, D, '0±15'), angle(IMAGE_A, F, C, '0±15'),
  dr_xy(IMAGE_A, E, F, south).

class(staircase, IMAGE_A, PLANESET_B) :-
  n_of_parts(IMAGE_A, PLANESET_B, 4),
  member(C, PLANESET_B), distributed_along(IMAGE_A, C, axisX).
```

The first rule defines PLANESET_B as a staircase in IMAGE_A if PLANESET_B has two planes C and D that are approximately parallel. PLANESET_B also contains plane E, which is distributed along the X-axis and is approximately perpendicular to planes C and D. The second rule covers plane sets that have planes C and D parallel to each other, E and F parallel to each other, with C and F also parallel and the directional relationship between E and F from the front view is south. Finally, the third rule represents plane sets having four planes distributed along axis X.

One of the most useful attributes of ILP is that learned concepts can become background knowledge for later training, thus allowing the system to build complex hierarchical representations. For example, a ‘staircase’ may be described as a set of planes

but a more general description might consider it to be an ordered set of steps, where the concept of "step" has been previously learned. We investigated this idea to see how relational representations can be used to accumulate knowledge. To do that, we added the previously learned description of class *step* to the background knowledge while learning *staircase*. The learned description of *staircase* now includes the *step*:

```
class(staircase, IMAGE_A, B) :-
  member(C, B), ratio_xz(IMAGE_A, C, '5.0±0.25'),
  has_class(step, IMAGE_A, D, B),
  has_class(step, IMAGE_A, E, B),
  intersect(D, E).
```

This rule says that *planeset*, B, in image IMAGE_A is staircase if B has a plane, C, where the ratio between its distribution along the X-axis and the Z-axis is 5.0±0.25 and B contains *planesets*, D and E, which are both steps and they intersect each other, meaning that they have at least one plane in common.

6 Conclusion

This paper demonstrates the ability of ILP to learn relational representations of object classes from 3D point clouds. By using the plane as a primitive component, a point cloud is segmented using point–based surface normal vectors. Plane features and plane-pair relationships, such as the angle between planes and their directional relationships, are used to convert the input data into training examples for ALEPH, an ILP learning system. 10-fold cross validation indicates that this approach is capable of producing highly accurate classifiers.

The region growing algorithm can benefit from a noise reduced point cloud, since the normal vector calculation and the region growing algorithm are sensitive to the noise and might produce incorrectly merged regions. Noise reduction algorithms such as jump edge filtering [22] may be suitable, especially for finding better boundaries [23-25] for each region.

We would like to learn to recognise a greater variety of objects, common in RoboCup Rescue such as barrels and ramps, as well as objects in a home and office environments, extending the method to more domains.

More experiments are needed to demonstrate that the learned concepts are robust to variations in scale and rotation. Some modifications of the features could be useful in this regard. For example, the feature `ch_ratio(image, plane, ratiobin)`, gives the ratio between diameter and width of a region's convex hull. This feature might appear in some rules with different `ratiobin` values for the same object class, e.g. '1±0.25', '1.5±0.25' and '2±0.25'. If we introduce an *interval* for `ratiobin`, the three values can be represented as [1-0.25,2+0.25].

The system can be modified to operate in an unsupervised learning mode, where the user does not need to label plane sets. Instead, we use CAD models to extract features for each object class as suggested by Böhm et al. in [26].

We have tested our algorithm using Microsoft Xbox Kinect range camera. However, the algorithm is parameterised so that it can be used on other range data (e.g. obtained with the SwissRanger-SR3000). These parameters, threshold values and bin sizes can be learned, rather than having them defined by the user.

In this work, we have chosen to build one binary classifier for each class. We would like to compare this against building a single multi-class classifier and how this affects the performance [27] .

References

1. Gachter, S.: Results on Range Image Segmentation for Service Robots (Technical Report). (2005)
2. Gachter, S., Nguyen, V., Siegwart, R.: Results on Range Image Segmentation for Service Robots. In: IEEE International Conference on Computer Vision Systems, pp. 53-53. (2006)
3. Hegazy, D., Denzler, J.: Generic 3D Object Recognition From Time-of-Flight Images Using Boosted Combined Shape Features. In: Proceedings of International Conference on Computer Vision, Theory and Applications (VISAPP). (2009)
4. Opelt, A.: Generic Object Recognition. Institute of Electrical Measurement and Measurement Signal Processing, vol. PhD. Graz University of Technology (2006)
5. Vasudevan, S., Gächter, S., Nguyen, V., Siegwart, R.: Cognitive maps for mobile robots--an object based approach. Robotics and Autonomous Systems 55, 359-371 (2007)
6. Shin, J.: Parts-Based Object Classification for Range Images. Autonomous Systems Lab, vol. Master Degree, pp. 48. Swiss Federal Institute of Technology Zurich (2008)
7. Endres, F.L.: Scene Analysis from Range Data. Department of Computer Science, Autonomous Intelligent Systems Lab, vol. Master. Albert-Ludwigs-University Freiburg, Faculty of Applied Sciences (2009)
8. RoboCup,RoboCup Rescue League, http://www.robocup.org/robocup-rescue/
9. Kadous, M.W., Sammut, C., Sheh, R.K.M.: Behavioural cloning for robots in unstructured environments. In: Workshop on Machine Learning based Ground Robotics, Neural Information Processing Systems. (2005)
10. Lowe, D.G.: Object recognition from local scale-invariant features. Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on 2, 1150-1157 (1999)
11. Fergus, R., Perona, P., Zisserman, A.: Object class recognition by unsupervised scale-invariant learning. In: Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 264-271. (2003)
12. Froimovich, G., Rivlin, E., Shimshoni, I., Soldea, O.: Efficient search and verification for function based classification from real range images. Computer Vision and Image Understanding 105, 200-217 (2007)
13. Shanahan, M.: A logical account of perception incorporating feedback and expectation. In: Proceedings KR, pp. 3-13. (2002)
14. Shanahan, M., Randell, D.: A logic-based formulation of active visual perception. In: Proceedings KR, pp. 64–72. (2004)
15. Srinivasan, A.,University of Oxford, http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/aleph.html
16. MESA,SwissRanger™ SR-3000, http://www.mesa-imaging.ch/prodviews.php
17. NIST,Test Methods, http://www.nist.gov/el/isd/test-methods.cfm

18. Vince, J.A.: Geometry for Computer Graphics: Formulae, Examples and Proofs. SpringerVerlag (2005)
19. Peuquet, D.J., Ci-Xiang, Z.: An algorithm to determine the directional relationship between arbitrarily-shaped polygons in the plane. Pattern Recognition 20, 65-74 (1987)
20. Dönderler, M., Ulusoy, Ö., Güdükbay, U.: A Rule-Based Approach to Represent Spatio-Temporal Relations in Video Data. In: Yakhno, T. (ed.) Advances in Information Systems, vol. 1909, pp. 409-418. Springer Berlin / Heidelberg (2000)
21. Witten, I.H., Frank, E., Hall, M.A.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann (2011)
22. Holz, D., Schnabel, R., Droeschel, D., Stückler, J., Behnke, S.: Towards Semantic Scene Analysis with Time-of-Flight Cameras. In: Ruiz-del-Solar, J., Chown, E., Plöger, P. (eds.) RoboCup 2010: Robot Soccer World Cup XIV, vol. 6556, pp. 121-132. Springer Berlin / Heidelberg (2011)
23. Cang, Y., Hegde, G.P.M.: Robust edge extraction for SwissRanger SR-3000 range images. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 2437-2442. (2009)
24. Sotoodeh, S.: Outlier detection in laser scanner point clouds. In: International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI-5, pp. 297-302. (2006)
25. Weber, C., Hahmann, S., Hagen, H.: Methods for Feature Detection in Point Clouds. In: Visualization of Large and Unstructured Data Sets - Applications in Geospatial Planning, Modeling and Engineering (IRTG 1131 Workshop), pp. 90-99. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, (2011)
26. Böhm, J., Brenner, C.: Curvature based range image classification for object recognition. In: Proceedings of Intelligent Robots and Computer Vision XIX: Algorithms, Techniques, and Active Vision, pp. 211-220. (2000)
27. Abudawood, T., Flach, P.: Learning Multi-class Theories in ILP. In: Frasconi, P., Lisi, F. (eds.) 20th International Conference Inductive Logic Programming, vol. 6489, pp. 6-13. Springer Berlin / Heidelberg (2011)