Mach Learn
DOI 10.1007/s10994-013-5352-9

# Plane-based object categorisation using relational learning

**Reza Farid · Claude Sammut**

Received: 15 December 2012 / Accepted: 9 April 2013

**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 categorisation 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 results is evaluated by 10-fold cross validation. In addition, common household objects that have curved surfaces are used for evaluation and comparison against a well-known non-relational classifier. The results show that ILP can be successfully applied to recognise objects encountered by a robot especially in an urban search and rescue environment.

**Keywords** Object classification · Inductive logic programming · Machine learning · Urban search and rescue · 3D point cloud

## 1 Introduction

In this work, we 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 the representation of classes of objects and a classification method to recognise new objects. Object features may be visual, structural, functional, etc.

Editors: Fabrizio Riguzzi and Filip Zelezny.

R. Farid (✉) · C. Sammut
School of Computer Science and Engineering, The University of New South Wales, Sydney, NSW 2052, Australia
e-mail: rezaf@cse.unsw.edu.au

C. Sammut
e-mail: claude@cse.unsw.edu.au

Published online: 26 April 2013

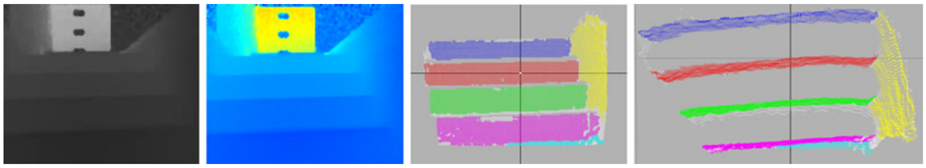


**Fig. 1** Range image and its correspondent point cloud (*coloured*) from front and top view (Color figure online)

3D depth cameras, such as the Microsoft Xbox Kinect and ASUS Xtion PRO LIVE, are now becoming widely used because they provide both range and video images and their cost are 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 (Gachter 2005; Gachter et al. 2006). This can be used to infer the shape of the object (Hegazy and Denzler 2009). These sensors, also incorporate 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 (Opelt 2006). A range image can be transformed into a set of 3D coordinates for each pixel, producing a point cloud. Figure 1 shows a range image of a staircase with four steps, taken by a robot positioned in front of the staircase. In this grey scale image, the darker colour represents closer surfaces. For more clarity, a colour-mapped version is also presented. The range image is converted into a point cloud. Some points in this cloud are removed because they are far away such as the points in the yellow region of the colour-mapped version. The figure also includes front and top views for the same point cloud. The point cloud is 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 indoor urban search and rescue for identifying floors, walls, staircases, ramps and other terrain that the robot is likely to encounter. For outdoor environments, other primitives may be used. However, the emphasis in this paper is object recognition in indoor environments. An ILP system is used to discover the properties and relationships between the planes that form an object and to represent them as logical rules, based on the training examples and background knowledge (Muggleton 1991). In the following sections, we describe the plane extraction method, the learning algorithm and the experimental results that demonstrate the utility of this approach.

An earlier version of this paper was presented at 22nd International Conference on Inductive Logic Programming (Farid and Sammut 2012b). In this paper, we provide more details and, in the earlier version, we had not considered noise tolerance. We have performed an entirely new set of experiments for this revised paper, including noise handling. We have also extended feature selection, examining prior filtering of redundant predicates and using measurements such as the amount of compression and the number of rules in our evaluations. We also add new comparisons between ALEPH (Srinivasan 2001) and Metagol (Stephen Muggleton et al. 2013). We show the usefulness of ILP's human-readable representation and also test the robustness of learning for a different camera and variations of object shapes. Finally, we compare the accuracy of our method with a well-known non-relational object classifier on images of some common household objects.

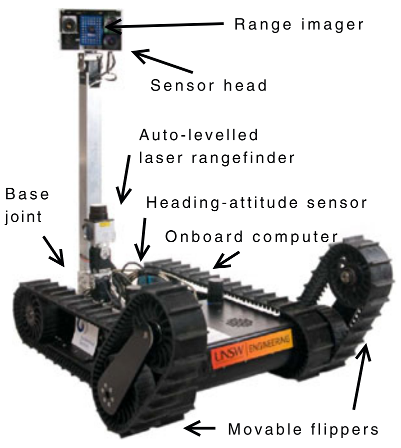


**Fig. 2** The rescue robot platform

## 2 Background and related work

A considerable amount of research has been devoted to generic object recognition (Opelt 2006; Vasudevan et al. 2007; Shin 2008; Endres 2009), which is required by robots in many tasks. For example in service robotics applications, such as a catering or a domestic robot (Shin 2008), the robot must recognise specific kinds of tableware. In industrial applications, the robot has to distinguish a set of products (Endres 2009). We are mostly interested in urban search and rescue (USAR), where a team of robots is sent to a disaster site. The robots are intended to search for victims and return to the human rescuers about the location and condition of the victims. The RoboCup Rescue Robot competition is a test bed for research in USAR (Kadous et al. 2005) and is the source of most of the training data used in our experiments.

In recent years, statistical methods such as SIFT (Lowe 1999) have become popular in 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) (Fergus et al. 2003), the system learns to recognise an object as belonging to a generic class (Froimovich et al. 2007), as one would expect in concept learning. A generic description of an object class has been suggested as a combination of structural and functional concepts extracted from input data by Froimovich et al. (2002). Pechuk et al. (2008) have developed a function-based object classification method. In addition, Posner et al. (2007, 2008) suggest a semantic labelling system for outdoor urban workspaces that uses SVMs for classification. Although, SVMs can be used for object recognition, they cannot provide information about an object's structure, which is needed for robot action planning, as we explain later. In contrast, relational learning is well suited for learning object classes, provided that the primitive features needed for recognition can be reliably extracted from the image. Xu and Petrou (2010) have used Markov logic networks for scene interpretation and categorisation.

Our approach is most closely related to Shanahan (2002), Shanahan and Randell (2004) 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 (Srinivasan 2001) to learn the logic programs from instances obtained by a robot equipped with a depth camera. Originally this was a SwissRanger SR-3000 camera, which has now been replaced by a Kinect. The

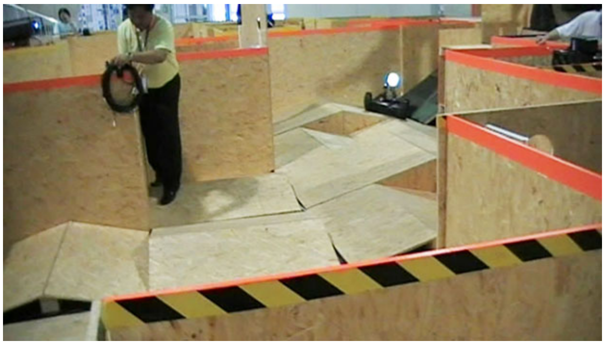


**Fig. 3** Roll and pitch ramps in a RoboCup rescue arena

robot is shown in Fig. 2. It was designed to participate in the RoboCup Rescue Robot competition, held annually. The competition arena uses elements designed by the US National Institute of Standards and Technology (NIST 2010) to certify robots for emergency operations. 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.

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 characterised by relationships between their 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 required for feature extraction.

## 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 which is based on the angle between them as explained in our previous work (Farid and Sammut 2012a). Figure 1 shows an example of planes found using this method. Attributes of the planes are then calculated, including the spherical coordinate 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 planes are labelled according to the class to which they belong. The ALEPH ILP system (Srinivasan 2001) 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 coordinate representation of a plane's normal vector ($\theta$ and $\varphi$) (Vince 2005), where $\theta$ is defined as zero for undefined situations, other attributes are

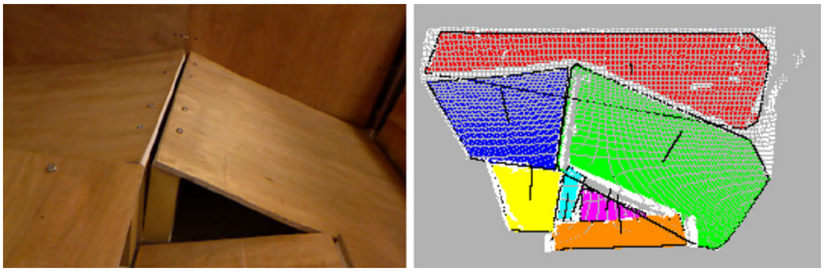
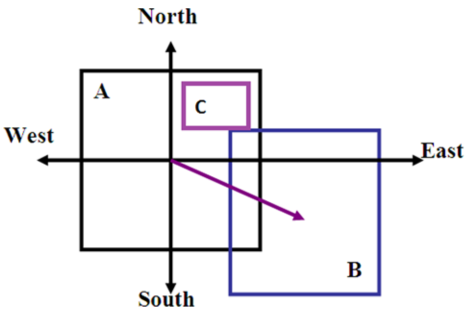


**Fig. 4** *Colour* image and segmented point cloud (front view), representing convex hull and normal vector for each region (Color figure online)

**Fig. 5** East (B, A), covers (A, C), connected (B, C)



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. Figure 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 adjacent planes. The second is a directional relationship (Peuquet and Ci-Xiang 1987; Dönderler et al. 2000) 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, from the perspective one 2D view. Also rectangle A covers C, while rectangles B and C are connected. Since planes exist in 3D 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.

## 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

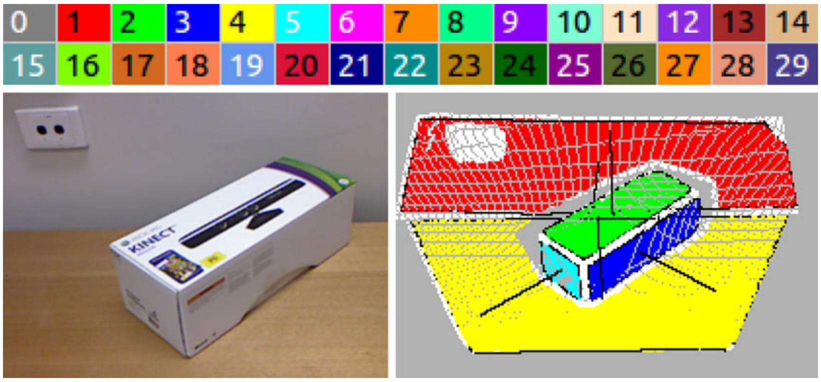


**Fig. 6** *Colour* legend (*top*), *colour* image and segmented point cloud (front view) for box (Color figure online)

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. Each image may contain several objects, which the system will learn to distinguish and recognise. For example, Fig. 4 contains a 'pitch/roll ramp' and a maze 'wall', and Fig. 6 contains three classes as 'wall', 'desk-top' and 'box'. 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 and the Asus Xtion PRO LIVE.

After labelling, the data set is ready to be used for learning. The labelled planes are represented as a set of Prolog ground clauses. Figure 6 shows the legend for the colours used, the point cloud segmentation with each region's convex hull and normal vector and the corresponding colour image. 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 the class "box" that includes regions 2, 3 and 5, creating the predicate $box([pl00\_2, pl00\_3, pl00\_5])$. That is, predicates of the form $box(+plane\text{-}set)$ represent a set of planes in an image that form an instance of the class.

A training instance is described by a set of predicates, where a subset of predicates specifies which planes constitute the example. Note that, to make each plane identifier unique, we use a name based on the image number and plane number.

$$plane(pl00\_1). \qquad plane(pl00\_2). \qquad plane(pl00\_3).$$
$$plane(pl00\_4). \qquad plane(pl00\_5).$$

The next set of predicates describes the individual attributes for each plane. The first attribute is on which axis the plane is distributed most. To calculate this, we find the difference between the maximum and minimum values of a region's point coordinates ($\Delta x$, $\Delta y$ and $\Delta z$) and compare them to decide which axis should be chosen.

$$distributed\_along(pl00\_1, axisX).$$
$$distributed\_along(pl00\_2, axisX).$$

$$distributed\_along(pl00\_3, axisX).$$

$$distributed\_along(pl00\_4, axisX).$$

$$distributed\_along(pl00\_5, axisY).$$

We also use the ratio between each pair of values ($\Delta x/\Delta y$, $\Delta y/\Delta z$ and $\Delta x/\Delta z$) as another set of features. However, instead of using the exact value, we bin them in intervals defined around fixed values $1, 1.5, 2, 2.5, \ldots, 10$ and 10.5 ($\pm 0.25$). If a ratio is bigger than a maximum value (10.25 here), we represent it as '$10.5 \pm 0.25$'. For example, '1.23' and '1.12' both fall in the interval '$1 \pm 0.25$'. A similar discretisation is applied to another plane feature, the ratio between the diameter and width of the convex hull. To avoid having ratio values such as '0.005', we defined negative ratio values. For example, for plane *pl00_5*, $\Delta x < \Delta z$, therefore the ratio $\Delta x/\Delta z$ is represented as negative value of $\Delta z/\Delta x$. Bin ratios '$1 - 0.25$' and '$-1 + 0.25$' are not possible based on our definition. However, we kept the same '$\pm 0.25$' interval for all bins.

$$ratio\_yz(pl00\_1, `-1.0 \pm 0.25'). \qquad ratio\_xz(pl00\_1, `4.5 \pm 0.25').$$

$$ratio\_xy(pl00\_1, `5.0 \pm 0.25'). \qquad \ldots$$

$$ratio\_yz(pl00\_5, `1.0 \pm 0.25'). \qquad ratio\_xz(pl00\_5, `-1.5 \pm 0.25').$$

$$ratio\_xy(pl00\_5, `-1.5 \pm 0.25').$$

$$ch\_ratio(pl00\_1, `4.0 \pm 0.25'). \qquad ch\_ratio(pl00\_2, `2.5 \pm 0.25').$$

$$ch\_ratio(pl00\_3, `3.5 \pm 0.25'). \qquad ch\_ratio(pl00\_4, `2.0 \pm 0.25').$$

$$ch\_ratio(pl00\_5, `1.5 \pm 0.25').$$

The last plane attribute is the spherical coordinate representation of its normal vector. Similar to the ratio representation, we use an interval centred on a particular angle. For example, 91.35 and 87.87 are binned in the interval '$90 \pm 15$'.

$$normal\_spherical\_theta(pl00\_1, `-90 \pm 15').$$

$$normal\_spherical\_phi(pl00\_1, `135 \pm 15').$$

$$\ldots$$

$$normal\_spherical\_theta(pl00\_5, `-135 \pm 15').$$

$$normal\_spherical\_phi(pl00\_5, `112 \pm 15').$$

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, the relations are:

$$angle(pl00\_1, pl00\_2, `90 \pm 15'). \qquad angle(pl00\_1, pl00\_3, `45 \pm 15').$$

$$angle(pl00\_1, pl00\_4, `90 \pm 15'). \qquad angle(pl00\_1, pl00\_5, `45 \pm 15').$$

$$angle(pl00\_2, pl00\_3, `90 \pm 15'). \qquad angle(pl00\_2, pl00\_4, `0 \pm 15').$$

$$angle(pl00\_2, pl00\_5, `90 \pm 15'). \qquad angle(pl00\_3, pl00\_4, `90 \pm 15').$$

$$angle(pl00\_3, pl00\_5, `90 \pm 15'). \qquad angle(pl00\_4, pl00\_5, `90 \pm 15').$$

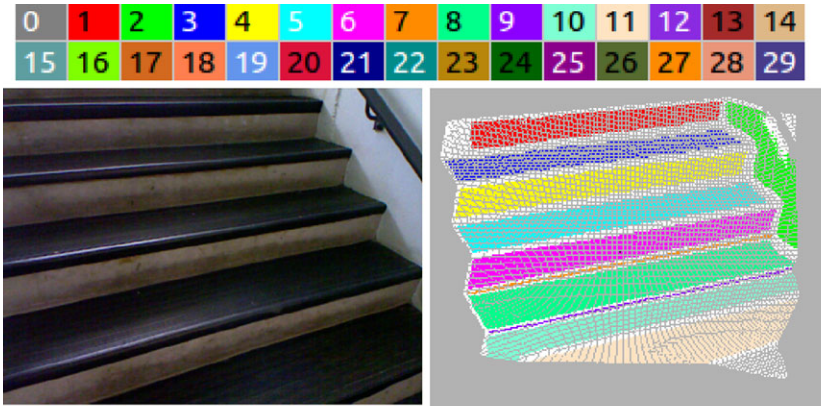


**Fig. 7** Stairs with different number of planes

$dr\_xz(pl00\_1, pl00\_2, connected).$ $dr\_xz(pl00\_1, pl00\_2, west).$

$dr\_xz(pl00\_2, pl00\_1, east).$ $dr\_xz(pl00\_2, pl00\_3, west).$

...

$dr\_xz(pl00\_5, pl00\_3, connected).$ $dr\_xz(pl00\_5, pl00\_3, south).$

$dr\_xz(pl00\_5, pl00\_4, covers).$

$dr\_xy(pl00\_1, pl00\_2, connected).$ $dr\_xy(pl00\_1, pl00\_2, north).$

$dr\_xy(pl00\_1, pl00\_3, connected).$ $dr\_xy(pl00\_1, pl00\_3, north).$

...

$dr\_xy(pl00\_4, pl00\_5, is\_covered).$ $dr\_xy(pl00\_5, pl00\_2, east).$

$dr\_xy(pl00\_5, pl00\_4, covers).$

Regions 2, 3 and 5, which form a box, are perpendicular to each other and this is represented by their pair-wise angle relationships.

We use planes 1 and 2 to illustrate a directional relationship. From both views, front and top, the regions overlap. Thus, "*connected*" is the value of the third argument in both directional relationships. From the XY view, region 2 (green) is below region 1 (red) and plane1 is north of plane2, giving, $dr\_xy(pl00\_1, pl00\_2, north)$. 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 $dr\_xz(pl00\_1, pl00\_2, west)$.

The number of planes that form an object may differ. For example, different sets of planes from an image can form positive examples for staircase (Fig. 7):

$staircase([pl02\_06, pl02\_08, pl02\_10, pl02\_11]).$

$staircase([pl02\_05, pl02\_06, pl02\_08, pl02\_10]).$

$staircase([pl02\_04, pl02\_05, pl02\_06, pl02\_08]).$

$staircase([pl02\_01, pl02\_03, pl02\_04, pl02\_05]).$

$staircase([pl02\_01, pl02\_03, pl02\_04, pl02\_05, pl02\_06, pl02\_08]).$

$staircase([pl02\_03, pl02\_04, pl02\_05, pl02\_06]).$

**Table 1** Results for 10-fold cross validation, no noise accepted

| Object | No. positive | No. negative | Accuracy = (TP+TN)/N | Precision = TP/(TP+FP) | Recall = TP/(TP+FN) |
|---|---|---|---|---|---|
| Step | 197 | 718 | 95.74 | 92.47 | 87.31 |
| Staircase | 237 | 656 | 99.66 | 98.75 | 100 |
| Wall | 105 | 803 | 98.79 | 97.96 | 91.43 |
| Box | 143 | 771 | 96.28 | 94.31 | 81.12 |
| Pitch/roll ramp | 131 | 201 | 97.59 | 96.95 | 96.95 |
| mean ± std. (percentage) | | | 97.61 ± 1.65 | 96.09 ± 2.63 | 91.36 ± 7.54 |

**Table 2** Results for 10-fold cross validation, some noise accepted

| Object | No. positive | No. negative | Accuracy | Precision | Recall |
|---|---|---|---|---|---|
| Step | 197 | 718 | 95.63 | 89.05 | 90.86 |
| Staircase | 237 | 656 | 99.33 | 98.33 | 99.16 |
| Wall | 105 | 803 | 97.58 | 87.39 | 92.38 |
| Box | 143 | 771 | 95.84 | 85.23 | 88.81 |
| Pitch/roll ramp | 131 | 201 | 97.89 | 95.59 | 99.24 |
| mean ± std. (percentage) | | | 97.25 ± 1.54 | 91.12 ± 5.59 | 94.09 ± 4.83 |

$$staircase([pl02\_01, pl02\_03, pl02\_04, pl02\_05, pl02\_06, pl02\_08, pl02\_10]).$$

$$staircase([pl02\_04, pl02\_05, pl02\_06, pl02\_08, pl02\_10, pl02\_11]).$$

$$staircase([pl02\_01, pl02\_03, pl02\_04, pl02\_05, pl02\_06, pl02\_08, pl02\_11]).$$

## 5 Evaluation

### 5.1 Initial evaluation

To evaluate the learning system, which we call PLOCRL (PLane-based Object Categorisation using Relational Learning), we use 10-fold cross-validation. In one experiment, we do not allow any rule to cover any negative example, while in the second experiment, some false positives are accepted. There is a parameter in ALEPH for this purpose that can be set by "set(noise, +V)". "V" is an integer value that defines the upper bound on the number of false positives allowed and we set it to 10 in our experiment. However, none of the rules covered more than 8 false positives in practice. The performance of the learning algorithm is measured by accuracy, precision and recall as shown in Table 1 and Table 2.

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 that are, as much as possible, invariant to transforms and thus, enable the learning algorithm to find general descriptions.

An example of a learned classifier for the concept of "staircase" assuming no false positives are allowed, is shown below. It was constructed from 237 positive examples and 656 negative examples.

[Rule 1] [Pos cover = **186** Neg cover = 0]

$$\begin{array}{ll} \multicolumn{2}{l}{\boldsymbol{staircase(B)} :-} \\ member(C, B), & member(D, B), \\ dr\_xz(D, C, east), & member(E, B), \\ angle(E, C, \text{‘}0 \pm 15\text{’}), & dr\_xy(E, D, south). \end{array}$$

[Rule 2] [Pos cover = **213** Neg cover = 0]

$$\begin{array}{ll} \multicolumn{2}{l}{\boldsymbol{staircase(B)} :-} \\ member(C, B), & member(D, B), \\ angle(D, C, \text{‘}0 \pm 15\text{’}), & member(E, B), \\ angle(E, D, \text{‘}90 \pm 15\text{’}), & angle(E, C, \text{‘}90 \pm 15\text{’}), \\ distributed\_along(E, axisX). & \end{array}$$

[Rule 3] [Pos cover = **127** Neg cover = 0]

$$\begin{array}{ll} \multicolumn{2}{l}{\boldsymbol{staircase(B)} :-} \\ member(C, B), & member(D, B), \\ dr\_xy(D, C, south), & member(E, B), \\ angle(E, D, \text{‘}0 \pm 15\text{’}), & angle(E, C, \text{‘}0 \pm 15\text{’}). \end{array}$$

In these rules, the set of planes that constitute an object is denoted by variable 'B'. Thus, member(X, B) means X is a plane from plane set B.

The first rule defines plane set B as a staircase if it has two planes C and D that D is to the east of C in the XZ-view. It also contains plane E, which is approximately parallel to plane C. Also, the spatial-directional relationship between planes E and D in the XY-view is south. This rule covers 186 positive examples (above 78.48 % of all positive examples) and no negative examples.

The second rule defines B as a 'staircase' if B has planes C, D parallel to each other and E is distributed mostly along X-axis and perpendicular to C and D. This rule covers 213 positive examples (above 89.87 % of total positives) and no negative examples.

Finally, the third rule represents plane sets having at least three planes C, D and E where E is parallel to C and D, while D is to the south of C in the XY-view. This rule covers more than 53.58 % of the positive examples.

Classifiers learned when allowing some false positives are mostly shorter. For example, in the case of "staircase", three rules are induced. Two of these rules are similar to rule 2 and 3 above. However rule 1 changes, as shown below, with more positive example coverage:

**Table 3** Comparing compression average with and without noise tolerance

| Object | Without Noise Tolerance | With Noise Tolerance |
|---|---|---|
| step | 16.33 | 32.6 |
| staircase | 170.00 | 173.67 |
| wall | 11.25 | 11.25 |
| box | 09.06 | 15.43 |
| Pitch/roll ramp | 18.5 | 49 |

**Table 4** Comparing the number of rules with and without noise tolerance

| Object | Without Noise Tolerance | With Noise Tolerance |
|---|---|---|
| step | 15 | 10 |
| staircase | 3 | 3 |
| wall | 8 | 8 |
| box | 17 | 14 |
| Pitch/roll ramp | 12 | 4 |

[Rule 1] [Pos cover = **204** Neg cover = 7]

$$\boldsymbol{staircase}(\boldsymbol{B}) : -$$

$$\begin{array}{ll} member(C, B), & member(D, B), \\ dr\_xy(D, C, south), & member(E, B), \\ distributed\_along(E, axisX), & dr\_xy(E, D, south). \end{array}$$

To compare the results obtained when noise is tolerated or not, we consider the number of rules induced, the number of predicates for each rule, and finally the number of positive and negative examples covered by each rule. We use the compression measure taken from ALEPH, P-N-L+1 where P and N are the number of positive and negative examples covered by the rule and L is the number of literals. The average value of this measure and the number of rules are shown in Table 3 and Table 4 for each object class for two experiments mentioned. These tables show that allowing some tolerance to noise gives shorter rules, fewer rules and more compression and greater coverage. Thus, accepting some false positives is preferred.

In many cases, it is difficult to interpret and label part of an image without understanding its context (Esposito and Malerba 2001). ILP's ability to create relational concept descriptions can assist in learning such context dependencies. One of the most useful attributes of ILP is that learned concepts can become background knowledge for later learning, 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 be that a staircase is an ordered set of steps, where the concept of "step" has been previously learned. A recursive definition of staircase would allow a variable number of steps.

We first used ALEPH to construct concepts using additional background knowledge. The concept "step" was first learned, requiring 9 rules, 4 of them had less than 10 % positive coverage. Thus, we only show the 5 remaining rules here:

[Rule 1] [Pos cover = **115** Neg cover = **8**]

$$\boldsymbol{step(B)} :-$$

$$n\_of\_parts(B, 2),\quad member(C, B),$$
$$member(D, B),\quad dr\_xy(C, D, north),$$
$$normal\_spherical\_theta(C, \text{`}-90 \pm 15\text{'}),$$
$$distributed\_along(C, axisX),\quad distributed\_along(D, axisX).$$

[Rule 3] [Pos cover = **128** Neg cover = **1**]

$$\boldsymbol{step(B)} :-$$

$$n\_of\_parts(B, 2),\quad member(C, B),\quad member(D, B),$$
$$normal\_spherical\_theta(C, \text{`}-90 \pm 15\text{'}),\quad angle(D, C, \text{`}90 \pm 15\text{'}),$$
$$distributed\_along(D, axisX).$$

[Rule 4] [Pos cover = **24** Neg cover = **1**]

$$\boldsymbol{step(B)} :-$$

$$n\_of\_parts(B, 2),\quad member(C, B),$$
$$member(D, B),\quad angle(D, C, \text{`}0 \pm 15\text{'}),\quad dr\_xz(D, C, east).$$

[Rule 7] [Pos cover = **24** Neg cover = **1**]

$$\boldsymbol{step(B)} :-$$

$$n\_of\_parts(B, 2),\quad member(C, B),$$
$$member(D, B),\quad normal\_spherical\_phi(D, \text{`}112 \pm 15\text{'}),$$
$$dr\_xy(C, D, connected),\quad dr\_xy(C, D, north).$$

[Rule 8] [Pos cover = **40** Neg cover = **0**]

$$\boldsymbol{step(B)} :-$$

$$n\_of\_parts(B, 2),\quad member(C, B),\quad member(D, B),$$
$$ratio\_xz(D, \text{`}10.5 \pm 0.25\text{'}),\quad dr\_xz(C, D, west).$$

We then ran two experiments. In the first experiment, we added all "step" rules to the background knowledge, while in the second experiment, we just added the rules that had at least 10 % coverage on positive examples—the 5 best "step" rules mentioned above—to the background knowledge.

The result of the first experiment was just one rule that defines "staircase" based on "step" as follows. This rule covers all positive examples and a small number of negative examples.

[Rule 1] [Pos cover = **237** Neg cover = **6**]

$$\boldsymbol{staircase(B)} :-$$

$$subset(C, B),\quad step(C),$$
$$subset(D, B),\quad step(D),\quad intersect(C, D).$$

**Table 5** Mean and std. (%) for 10-fold cross validation before and after disabling some predicates

| Experiment | Accuracy | Precision | Recall |
|---|---|---|---|
| 0 | $97.25 \pm 1.54$ | $91.12 \pm 5.59$ | $94.09 \pm 4.83$ |
| 1 | $96.87 \pm 2.41$ | $90.17 \pm 8.44$ | $93.4 \pm 6.35$ |
| 2 | $97.01 \pm 2.29$ | $91.25 \pm 7.68$ | $92.02 \pm 7.79$ |
| 3 | $97.16 \pm 2.41$ | $93.52 \pm 6.01$ | $90.49 \pm 12.3$ |

This rule says that plane set B is a staircase if it contains plane sets C and D, both of which are steps and intersect each other, meaning that they are different and have at least one plane in common.

The second experiment yields two rules. The first rule that uses "step" covered 235 out of 237 positive examples and 5 negative examples.

[Rule 1] [Pos cover = **235** Neg cover = **5**]

$$\boldsymbol{staircase(B)} :-$$
$$member(C, B),\quad member(D, B),\quad angle(C, D, \text{‘}0 \pm 15\text{’}),$$
$$distributed\_along(D, axisX),\quad subset(E, B),\quad step(E).$$

[Rule 2] [Pos cover = **7** Neg cover = **0**]

$$\boldsymbol{staircase(B)} :-$$
$$member(C, B),\quad ratio\_yz(C, \text{‘}8.0 \pm 0.25\text{’}),$$
$$ratio\_xz(C, \text{‘}9.0 \pm 0.25\text{’}).$$

Thanks to the assistance of Stephen Muggleton and Dianhuan Lin, we were able to compare the results from ALEPH with Metagol (Muggleton et al. 2013). Muggleton and Lin attempted to learn the recursive concept of staircase using a simplified version of the data set that just contains "angle" predicates. Metagol is capable of predicate invention and thus, the "p_a" predicate, which is the concept of a step, was created automatically. The clauses learned by Metagol are:

$$\boldsymbol{staircase(B)} :- p\_a(B).$$
$$\boldsymbol{staircase}([X, Y, Z|B]) :-$$
$$p\_a([X, Y, Z]),\quad staircase([Z|B]).$$
$$\boldsymbol{p\_a(B)} :-$$
$$member(X, B),\quad member(Y, B),$$
$$angle(X, Y, \text{‘}90 \pm 15\text{’}),\quad member(Z, B),\quad angle(X, Z, \text{‘}0 \pm 15\text{’}).$$

## 5.2 Feature evaluation

In this section, we describe three experiments with ALEPH in which we try to determine the most informative predicates and which of the less informative ones can be eliminated. In the first experiment, we remove the 2D spatial-directional predicates. In the second, we

**Table 6** Comparing compression average for two selected experiments

| Object | Experiment 1 | Experiment 2 |
|---|---|---|
| step | 32.6 | 13.29 |
| staircase | 173.67 | 144 |
| wall | 11.25 | 8.9 |
| box | 15.43 | 3.41 |
| Pitch/roll ramp | 49 | 50.33 |

also eliminate "ratio_yz", "ratio_xy", "ratio_xz" and "distributed_along". In the last experiment we remove "ch_ratio", only keeping "angle", "normal_spherical_theta" and "normal_spherical_phi". Using the same data, the 10-fold cross validation results are presented in Table 5, while experiment 0 shows the result when no predicate is removed. As there is no significant loss of accuracy, all the removed predicates are redundant. However, the length of the concept description increases while the average compression has decreased as shown in Table 6. In this table, we compare two learning runs: in the first, no predicate is removed and in the second, we keep the minimum set of predicates "angle", "normal_spherical_theta" and "normal_spherical_phi". So if simplicity of the description and compression are used as evaluation criteria, the eliminated predicates are not redundant, as they allow more compact descriptions to be learned. An example is shown below:

[Rule 1] [Pos cover = **143** Neg cover = **2**]

$$\boldsymbol{staircase(B)} :- $$

$$\begin{array}{ll} member(C, B), & member(D, B), \\ angle(D, C, \text{‘}0 \pm 15\text{’}), & member(E, B), \\ angle(E, D, \text{‘}0 \pm 15\text{’}), & angle(E, C, \text{‘}0 \pm 15\text{’}). \end{array}$$

[Rule 2] [Pos cover = **213** Neg cover = **1**]

$$\boldsymbol{staircase(B)} :- $$

$$\begin{array}{ll} member(C, B), & member(D, B), \\ angle(D, C, \text{‘}0 \pm 15\text{’}), & member(E, B), \\ angle(E, D, \text{‘}90 \pm 15\text{’}), & angle(E, C, \text{‘}90 \pm 15\text{’}). \end{array}$$

[Rule 3] [Pos cover = **92** Neg cover = **0**]

$$\boldsymbol{staircase(B)} :- $$

$$\begin{array}{ll} n\_of\_parts(B, 4), & member(C, B), \\ member(D, B), & angle(D, C, \text{‘}0 \pm 15\text{’}). \end{array}$$

The first rule defines plane set B as a staircase if B has at least three planes C, D and E that are approximately parallel.

The second rule considers plane set B to be a staircase if B contains three planes C, D and E, where C and D are parallel and both are perpendicular to E. This rule is similar to the invented predicate, "p_a", that appeared in the previous section using Metagol.

The third rule states that a plane set B having four planes is a staircase if it has two planes parallel to each other.

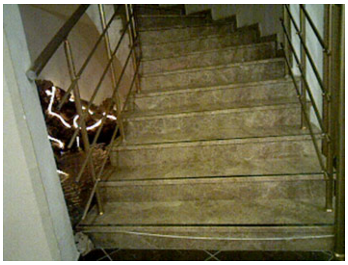


**Fig. 8** Spiral staircase (*colour* image) (Color figure online)

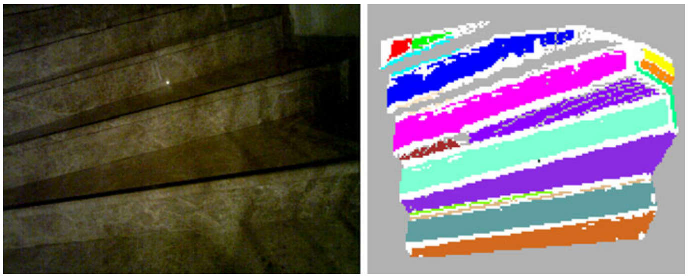


**Fig. 9** One image of spiral stairs and its point cloud segmentation result

An advantage of ILP is the readability of its output. This is useful for expert interpretation of the learned concept and also helpful to novices in a domain. For example, in our earlier training attempts, we used a smaller training set for "staircase" (237 positive examples and 637 negative examples). In that experiment, we had removed 2D spatial-directional relationship predicates, as in experiment 1 above. The learned classifier contained three rules, one of which was:

[Rule 3] [Pos cover = **92** Neg cover = **0**]

$$\boldsymbol{staircase(B)} :-$$
$$n\_of\_parts(B, 4).$$

That is, a set of four planes was considered to be a staircase. This indicated that the training set did not include negative examples that had four planes. By adding few such negative examples (six in practice), the more specific rule was induced as below. Thus, readability of the concept description, as obtained in ILP, facilitated analysis of the training data.

[Rule 3] [Pos cover = **92** Neg cover = **0**]

$$\begin{aligned}&\boldsymbol{staircase(B)} :-\\&\quad n\_of\_parts(B, 4),\\&\quad member(C, B), \qquad distributed\_along(C, axisX).\end{aligned}$$

## 5.3 Different camera and different training sets

For the third evaluation, we captured data using an ASUS Xtion PRO LIVE sensor, which is similar to the Kinect. We also checked the robustness of learning for variations of object shapes. For example, can a spiral staircase (Fig. 8) be recognised by rules trained on straight

**Table 7** Results for testing new data (spiral staircase) using existing rules

| Object | No. positive | No. negative | Accuracy | Precision | Recall |
|---|---|---|---|---|---|
| Staircase | 948 | 0 | 95.15 | 100 | 95.15 |

**Table 8** Results for 10-fold cross validation after adding spiral staircase to the data set

| Object | No. positive | No. negative | Accuracy | Precision | Recall |
|---|---|---|---|---|---|
| Staircase | 1185 | 656 | 99.51 | 99.25 | 100 |

**Table 9** Chosen subset of rgbd-dataset used for the last evaluation

| Object | No. positive | No. negative | No. of physically distinct instance sets | Number of images for each physically distinct instance set |
|---|---|---|---|---|
| ball | 276 | 1076 | 7 | [41, 39, 40, 40, 39, 39, 38] |
| bowl | 193 | 1159 | 6 | [29, 29, 29, 27, 40, 39] |
| cap | 126 | 1226 | 4 | [31, 33, 31, 31] |
| cereal box | 144 | 1208 | 5 | [28, 30, 30, 29, 27] |
| coffee mug | 240 | 1112 | 8 | [27, 27, 27, 27, 26, 27, 40, 39] |
| kleenex | 189 | 1163 | 5 | [38, 39, 40, 32, 40] |
| lemon | 184 | 1168 | 6 | [29, 33, 30, 30, 32, 30] |
| Total | 1352 | | | |

staircases and how is accuracy improved by including examples of spiral staircases. Figure 9 shows one scene with its point cloud segmentation. In these experiments, we accept some noise and the spatial-directional relationship predicates are eliminated. To test the robustness of the rules trained on mostly straight staircases, 948 new examples from a spiral staircase were classified. More than 95 % accuracy is obtained, as shown in Table 7. When these 948 examples from the spiral staircase are added to the data set and tested with 10-fold cross validation, the accuracy rises further, as shown in Table 8. Both results indicate the robustness of the classification method.

## 5.4 Comparison with a non-relational object classifier

In this experiment, we test the accuracy of our method on images of some common household objects collected using an RGB-D camera, organized into 51 categories and presented by Lai et al. (2011) (http://www.cs.washington.edu/rgbd-dataset). Each object in this dataset was placed on a turntable and a sequence of videos was taken for each complete rotation. This procedure was repeated three times with different camera positions to guarantee multiple views. We also would like to compare the result with a non-relational classifier (Bo et al. 2011) using the same dataset. For this experiment, we have chosen 7 categories 'ball', 'bowl', 'cap', 'cereal box', 'coffee mug', 'kleenex' and 'lemon'. Using the original captured and cropped images, we sub-sample each sequence by taking every 20th frame. Figure 10 shows some instances (coloured image version) of the chosen categories. Each image in the dataset is taken as a positive example for its category and a negative example for the rest. The number of physically distinct instances for each category and number of positive and negative examples used for this experiment are presented in Table 9.

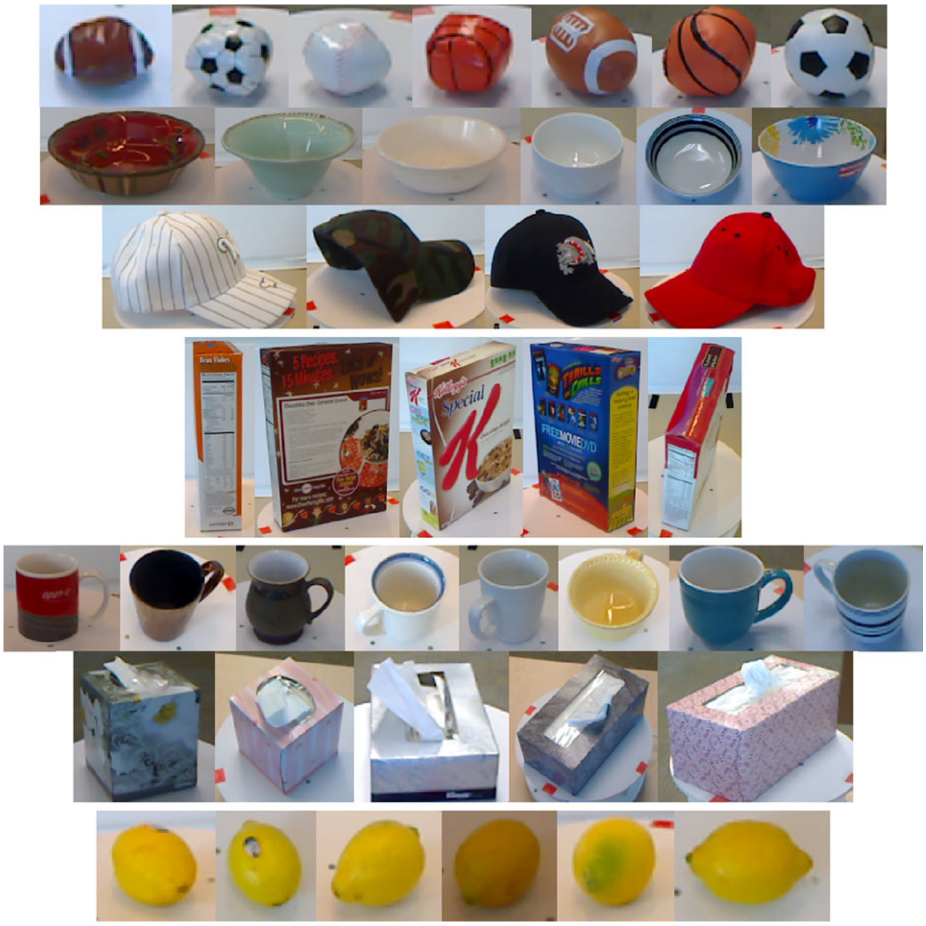


**Fig. 10** Seven categories with some physically distinct instance sets

In this evaluation, we use the same method as Bo et al. (2011), that is, 10 train/test splits that randomly choose one physically distinct instance set as the test set and the rest as the training set in each iteration. A physically distinct instance set is a collection of images from a particular type of object taken from different angles. We have used their MATLAB code.[1] However, we have modified their category recognition method based on the documentation they provided in their C++ code, i.e. using the 'linear' option for scaling the training and testing data. Since they have introduced some depth kernel descriptors for object classification using depth and colour images, we have used gradient kernel descriptors (*Gradient KDES*) and local binary pattern kernel descriptors (*LBP KDES*) separately. The average accuracy of classification using the above descriptors and our method are shown in Table 10. Similar to our previous evaluations, some false positives were accepted in our method.

[1] http://www.cs.washington.edu/ai/Mobile_Robotics/projects/kdes/.

**Table 10** Comparison to other method using rgbd-dataset

| Method | Accuracy (mean ± std.) (percentage) |
|---|---|
| Gradient KDES | 89.07 ± 5.10 |
| LBP KDES | 86.51 ± 3.10 |
| PLOCRL | 87.90 ± 0.91 |

**Table 11** Number of rules induced

| Object | Number of rules |
|---|---|
| ball | 30 |
| bowl | 27 |
| cap | 21 |
| cereal box | 7 |
| coffee mug | 36 |
| kleenex | 39 |
| lemon | 27 |

Although our method is based on 'plane' primitives, the accuracy is comparable to a state-of-the-art object classifier when it is tested on common objects that have curved surfaces. Moreover, our method describes the relationship between subparts which we claimed to be a useful feature of relational learning. In addition, our relational method is able to learn by using previously learned concepts in the background knowledge. These two properties are not present in other methods such as the depth kernel descriptors we used for comparison here.

Because we have only used 'plane' primitives, we expected lower accuracy for this experiment in comparison to our previous experiments, which used objects with mostly planar surfaces. Somewhat surprisingly, the accuracy is quite high. However the average number of rules induced for each object class for categorisation increased as shown in Table 11. Extending primitives to more shapes such as cylinders and spheres should lead to improvements in accuracy, but particularly should result more readable rules and faster learning.

## 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. Preliminary experiments have also been conducted with Metagol (Muggleton et al. 2013). 10-fold cross validation indicates that this approach is capable of producing highly accurate classifiers. Further experiments with Metagol are intended since this system is capable of predicate invention.

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 (Holz et al. 2011) may be suitable, especially for finding better boundaries (Cang and Hegde 2009; Sotoodeh 2006; Weber et al. 2011) for each region.

We would like to learn to recognise a greater variety of objects from the rescue environment, as well as objects in home and office environments, extending the method to more domains and objects with curved surfaces. Although we showed that our method can be applied to objects with curved surfaces, it will be more helpful to construct a variety features, such as generalised cylinders, spheres and other shape primitives, as well as planes for the pre-processing. By extending the primitives from planes to a larger set of primitives, the number of extracted primitives and consequently the complexity of the rules should be reduced.

We intend to perform further testing on data sets that contain more similar objects. Although we have learned 'kleenex' and 'cereal box' or 'ball' and 'lemon' categories, there is still some analysis to be done on discriminating categories that have similar shapes. Other features may help to distinguish them more clearly.

Some features, such as directional relationships, can be modified to be more suitable for 3D space. A hierarchical structure that combines the spatial configuration with other information (Antanas et al. 2012) may also be a fruitful area to study.

Other modifications of the features can also improve performance. For example, the feature *ch_ratio*(*plane, ratiobin*), gives the ratio between diameter and width of a region's convex hull. This feature might appear in some rules with different consecutive 'ratiobin' values for the same object class, e.g. '$2 \pm 0.25$', '$2.5 \pm 0.25$' and '$3 \pm 0.25$'. If we introduce an interval for '*ratiobin*', the three values can be represented as $[2-0.25, 3+0.25]$. As a result, the number of rules can be reduced. Another option is to add this kind of generalization into the learner.

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 and Brenner (2000). Another option is using previously segmented data, similar to those which we have used for the final experiment here.

The image pre-processing is parameterised so that it can be applied to a variety of range images. We have tested our system using a SwissRanger SR-3000, the Microsoft Xbox Kinect and ASUS Xtion PRO Live sensors. 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 multiclass classifier and how this affects the performance. For this purpose, we can compare our current one-vs.-rest approach, which Rifkin and Klautau (2004) claim is more accurate than other multi-class approaches. However, Abudawood and Flach (2011) question this technique for first-order learning and suggest forming one rule set by combining the rules.

**Acknowledgements** The authors thank Stephen Muggleton and Dianhuan Lin for their assistance in running Metagol on the staircase data.

## References

Abudawood, T., & Flach, P. (2011). Learning multi-class theories in ILP. In P. Frasconi & F. Lisi (Eds.), *Lecture notes in computer science: Vol. 6489. 20th international conference on ILP* (pp. 6–13). Berlin: Springer.

Antanas, L., van Otterlo, M., Mogrovejo, O., Antonio, J., Tuytelaars, T., & De Raedt, L. (2012). A relational distance-based framework for hierarchical image understanding. In *ICPRAM 2012—proceedings of the 1st international conference on pattern recognition applications and methods*, Algarve, Portugal, 6–8 Feb 2012 (Vol. 2, pp. 206–218).

Bo, L., Ren, X., & Fox, D. (2011). Depth kernel descriptors for object recognition. In *Proc. of IEEE/RSJ international conference on intelligent robots and systems (IROS), 2011*.

Böhm, J., & Brenner, C. (2000). Curvature based range image classification for object recognition. In *Proceedings of intelligent robots and computer vision XIX: algorithms, techniques, and active vision, 2000* (Vol. 4197, pp. 211–220).

Cang, Y., & Hegde, G. P. M. (2009). Robust edge extraction for SwissRanger SR-3000 range images. In *IEEE international conference on robotics and automation (ICRA)*, 12–17 May 2009 (pp. 2437–2442).

Dönderler, M., Ulusoy, Ö., & Güdükbay, U. (2000). A rule-based approach to represent spatio-temporal relations in video data. In T. Yakhno (Ed.), *Lecture notes in computer science: Vol. 1909. Advances in information systems* (pp. 409–418). Berlin: Springer.

Endres, F. L. (2009). *Scene analysis from range data*. Master, Albert-Ludwigs-University Freiburg, Faculty of Applied Sciences.

Esposito, F., & Malerba, D. (2001). Machine learning in computer vision. *Applied Artificial Intelligence*, *15*(8), 693–705. doi:10.1080/088395101317018546.

Farid, R., & Sammut, C. (2012a). *A relational approach to plane-based object categorisation*. Paper presented at the RSS 2012 workshop on RGB-D cameras [Online], University of Sydney, July 2012.

Farid, R., & Sammut, C. (2012b). Plane-based object categorisation using relational learning. In *Proceedings of the 22nd international conference on inductive logic programming (ILP2012)*, Dubrovnik, Croatia, 17–19 September 2012

Fergus, R., Perona, P., & Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Proceedings IEEE computer society conference on computer vision and pattern recognition, 2003* (Vol. 2, pp. 264–271).

Froimovich, G., Rivlin, E., & Shimshoni, I. (2002). Object classification by functional parts. In *Proceedings first international symposium on 3D data processing visualization and transmission, 2002* (pp. 648–655).

Froimovich, G., Rivlin, E., Shimshoni, I., & Soldea, O. (2007). Efficient search and verification for function based classification from real range images. *Computer Vision and Image Understanding*, *105*(3), 200–217.

Gachter, S. (2005). *Results on range image segmentation for service robots* (Technical Report).

Gachter, S., Nguyen, V., & Siegwart, R. (2006). Results on range image segmentation for service robots. In *IEEE international conference on computer vision systems, 2006* (p. 53).

Hegazy, D., & Denzler, J. (2009). 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*.

Holz, D., Schnabel, R., Droeschel, D., Stückler, J., & Behnke, S. (2011). Towards semantic scene analysis with time-of-flight cameras. In J. Ruiz-del-Solar, E. Chown, & P. Plöger (Eds.), *Lecture notes in computer science: Vol. 6556. RoboCup 2010: robot soccer world cup XIV* (pp. 121–132). Berlin: Springer.

Kadous, M. W., Sammut, C., & Sheh, R. K. M. (2005). Behavioural cloning for robots in unstructured environments. In *Workshop on machine learning based ground robotics, neural information processing systems, 2005*.

Lai, K., Bo, L., Ren, X., & Fox, D. (2011). A large-scale hierarchical multi-view RGB-D object dataset. In *Proc. of international conference on robotics and automation (ICRA), 2011*.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *The proceedings of the seventh IEEE international conference on computer vision* (Vol. 2, pp. 1150–1157).

Muggleton, S. (1991). Inductive logic programming. *New Generation Computing*, *8*(4), 295–318.

Muggleton, S., Lin, D., Pahlavi, N., & Tamaddoni-Nezhad, A. (2013). Meta-interpretive learning: application to grammatical inference. *Machine Learning*. Special issue on Inductive Logic Programming.

NIST (2010). *The National Institute of Standards and Technology; Test Methods*. http://www.nist.gov/el/isd/test-methods.cfm.

Opelt, A. (2006). *Generic object recognition*. Graz: Graz University of Technology.

Pechuk, M., Soldea, O., & Rivlin, E. (2008). Learning function-based object classification from 3D imagery. *Computer Vision and Image Understanding*, *110*(2), 173–191.

Peuquet, D. J., & Ci-Xiang, Z. (1987). An algorithm to determine the directional relationship between arbitrarily-shaped polygons in the plane. *Pattern Recognition*, *20*(1), 65–74. doi:10.1016/0031-3203(87)90018-5.

Posner, I., Schroeter, D., & Newman, P. (2007). Describing composite urban workspaces. In *IEEE international conference on robotics and automation*, 10–14 April 2007 (pp. 4962–4968). doi:10.1109/robot.2007.364244.

Posner, I., Schroeter, D., & Newman, P. (2008). Online generation of scene descriptions in urban environments. *Robotics and Autonomous Systems*, *56*(11), 901–914. doi:10.1016/j.robot.2008.08.009.

Rifkin, R., & Klautau, A. (2004). In defense of one-vs-all classification. *Journal of Machine Learning Research*, *5*, 101–141.

Shanahan, M. (2002). A logical account of perception incorporating feedback and expectation. In *Proceedings KR, 2002* (pp. 3–13).

Shanahan, M., & Randell, D. (2004). A logic-based formulation of active visual perception. In *Proceedings KR, 2004* (Vol. 4, pp. 64–72).

Shin, J. (2008). *Parts-based object classification for range images*. Zurich: Swiss Federal Institute of Technology.

Sotoodeh, S. (2006). Outlier detection in laser scanner point clouds. In *International archives of photogrammetry, remote sensing and spatial information sciences XXXVI-5* (pp. 297–302).

Srinivasan, A. (2001). *The Aleph manual* (Technical report). University of Oxford.

Vasudevan, S., Gächter, S., Nguyen, V., & Siegwart, R. (2007). Cognitive maps for mobile robots—an object based approach. *Robotics and Autonomous Systems*, *55*(5), 359–371.

Vince, J. A. (2005). *Geometry for computer graphics: formulae, examples and proofs*. Berlin: Springer.

Weber, C., Hahmann, S., & Hagen, H. (2011). 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)*, Dagstuhl, Germany, 2011 (Vol. 19, pp. 90–99). Wadern: Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik. doi:10.4230/OASIcs.VLUDS.2010.90.

Xu, M., & Petrou, M. (2010). Learning logic rules for scene interpretation based on Markov logic networks. In H. Zha, R.-i. Taniguchi, & S. Maybank (Eds.), *Lecture notes in computer science: Vol. 5996. Computer vision—ACCV 2009* (pp. 341–350). Berlin: Springer.