# Instance-based AMN Classification for Improved Object Recognition in 2D and 3D Laser Range Data

**Rudolph Triebel Richard Schmidt Óscar Martínez Mozos Wolfram Burgard**
Department of Computer Science, University of Freiburg
Georges-Koehler-Alle 79, 79110 Freiburg, Germany
{triebel, rschmidt, omartine, burgard}@informatik.uni-freiburg.de

## Abstract

In this paper, we present an algorithm to identify different types of objects from 2D and 3D laser range data. Our method is a combination of an instance-based feature extraction similar to the Nearest-Neighbor classifier (NN) and a collective classification method that utilizes associative Markov networks (AMNs). Compared to previous approaches, we transform the feature vectors so that they are better separable by linear hyperplanes, which are learned by the AMN classifier. We present results of extensive experiments in which we evaluate the performance of our algorithm on several recorded indoor scenes and compare it to the standard AMN approach as well as the NN classifier. The classification rate obtained with our algorithm substantially exceeds those of the AMN and the NN.

## 1 Introduction

In this paper, we consider the problem of identifying objects in 2D and 3D range measurements. We believe that the segmentation of these data into known object classes can be useful in many different areas, such as map registration, robot localization, object manipulation, or human-robot interaction. For example, a robot that is able to classify the 3D data acquired by a range sensor has a better capability of finding corresponding data points in different measurements. In this way, the scan registration can be carried out more reliably. Other application areas, in which object recognition is important, include seek-and-rescue tasks such as the one aimed by the RoboCup rescue initiative.

What makes the object detection especially hard is the fact that it involves both, the segmentation and the classification of the segments. To simultaneously solve this segmentation and classification problem, recently collective classification approaches have become a popular approach. They are based on the assumption that in many real-world domains spatial neighbors in the data tend to have the same labels. For example, Anguelov *et al.* [2005] formulate the problem of segmenting 3D range data into known classes as a supervised learning task and apply collective classification to solve the task. They use a technique called associative Markov networks (AMNs), which combines the concept of relational learning with collective classification.

So far, AMNs have been used only based on the log-linear model, which means that there is only a linear relationship between the extracted features and the weight parameters learned by the algorithm. This results in a classification algorithm that learns hyper-planes in the feature space to separate between the object classes. However, in many real-world applications the assumption of the linear separability of the features is often not justified. One way to overcome this problem may be to extend the log-linear model, but this is still subject to ongoing research. In this paper, we propose a different approach. By combining the ideas of instance-based classification with the AMN approach, we obtain a classifier that is more robust against the error introduced by the linear-separability assumption. We present a way to compute new feature vectors from a given set of standard features, i.e., we transform the original features, and show that these features are better suited for classification using the AMN approach.

This paper is organized as follows. The next section gives an overview of the work that has been presented so far in this area. Then we summarize the concepts of collective classification using associative Markov networks and describe shortly how learning and inference is done in AMNs. For a more detailed analysis we refer to the work by Taskar *et al.* [2004]. Then, a detailed description of our approach is presented. Finally, we present the results of experiments, which illustrate that our method provides better classifications than previous approaches.

## 2 Related Work

The problem of recognizing objects from 3D range data has been studied intensively in the past. Most of the approaches can be distinguished according to the features used. One popular approach are spin images [Johnson, 1997; de Alarcón *et al.*, 2002; Frome *et al.*, 2004], which are rotationally and translationally invariant local descriptors. Spin images are also used as features in the work presented here. Other types of 3D features include local tensors [Mian *et al.*, 2004], shape maps [Wu *et al.*, 2004], and multi-scale features [Li and Guskov, 2005]. Osada *et al.* [2001] proposed a 3D object recognition technique based on shape distributions. Whereas this approach requires a complete view of the objects, our

method can deal with partially seen objects. Additionally, Huber *et al.* [2004] present an approach for parts-based object recognition. This method provides a better classification because nearby parts that are easier to identify than others help to guide the classification. A similar idea of detecting object components has been presented by Ruiz-Correa *et al.* [2003], who introduced symbolic surface signatures. Another approach to object recognition proposed by Boykov and Huttenlocher [1999] is based on Markov random fields (MRFs). They classify objects in camera images by incorporating statistical relationships between nearby object parts. A relational approach using MRFs has been proposed by Limketkai *et al.* [2005]. The idea here is to exploit the spatial relationship between nearby objects, which in this case consist of 2D line segments. The work that is mostly related to the approach described in this paper is presented by Anguelov *et al.* [2005], in which an AMN approach is used in a supervised learning setting to classify 3d range data. In this paper, we combine this approach with techniques from instance-based classification to obtain improved classification results.

## 3 Range Scan Classification

Suppose we are given a set of $N$ data points $\mathbf{p}_1, \ldots, \mathbf{p}_N$ taken from a 2D or 3D scene and a set of $K$ *object classes* $C_1, \ldots, C_K$. A data point in this context may be a 3D scan point or a 2D grid cell of an occupancy grid. The following formulations are identical in these two cases.

For each data point $\mathbf{p}_i$, we are also given a feature vector $\mathbf{x}_i$. Later we will describe in detail how the features are defined in the context of this paper. The task is to find a *label* $y_i \in \{1, \ldots, K\}$ for each $\mathbf{p}_i$ so that all labels $y_i, \ldots, y_N$ are optimal given the features. The notion of optimality in this context depends on the classification method we choose. In the standard supervised learning approach, we define a likelihood function $P_\omega(\mathbf{y} \mid \mathbf{x})$ of the labels $\mathbf{y}$ given the features $\mathbf{x}$. Then, the classification problem can be subdivided into two sub tasks: first we need to find good parameters $\omega^*$ for the likelihood function $P_\omega(\mathbf{y} \mid \mathbf{x})$, then we seek for good labels $\mathbf{y}^*$ that maximize this likelihood. Assuming we are given a *training* data set, in which the labels $\hat{\mathbf{y}}$ have been assigned by hand, we can formulate the classification as follows:

- learning step: find $\omega^* = \text{argmax}_\omega P_\omega(\hat{\mathbf{y}} \mid \mathbf{x})$
- inference step: find $\mathbf{y}^* = \text{argmax}_\mathbf{y} P_{\omega^*}(\mathbf{y} \mid \mathbf{x})$

### 3.1 Collective Classification

In most standard classification methods, such as Bayes classification, Nearest Neighbor, AdaBoost etc. the classification of a data point only depends on its local features, but not on the labeling of nearby data points. However, in our setting there is a statistical dependence of the labelings associated to neighboring data points. For example, if we consider the local planarity of a scan point as a feature, it may happen that the likelihood $P(\mathbf{y} \mid \mathbf{x})$ of the class label 'wall' is higher than that of the class label 'door', although all other scan points in the vicinity of this point belong to the class 'door'. Methods that use the information of neighboring data points are denoted as *collective classification* techniques (see Chakrabarti and Indyk[1998]). Recently, a collective classification approach to classify 3D range scan data has been presented by Anguelov *et al.* [2005]. The authors propose the use of associative Markov networks (AMNs), which are a special type of Relational Markov Random Fields, described in Taskar *et al.*[2002]. In the following we will briefly describe AMNs.

### 3.2 Associative Markov Networks

An associative Markov network is an undirected graph in which the nodes are represented by $N$ random variables $y_1, \ldots, y_N$. In our case, these random variables are discrete and correspond to the labels of each of the data points $\mathbf{p}_1, \ldots, \mathbf{p}_N$. Each node $y_i$ and each edge $(y_i, y_j)$ in the graph has an associated non-negative value $\varphi(\mathbf{x}_i, y_i)$ and $\psi(\mathbf{x}_{ij}, y_i, y_j)$ respectively. These are also known as the *potentials* of the nodes and edges. The node potentials reflect the fact that for a given feature vector $\mathbf{x}_i$ some labels are more likely to be assigned to $\mathbf{p}_i$ than others, whereas the edge potentials encode the interactions of the labels of neighboring nodes given the edge features $\mathbf{x}_{ij}$. Whenever the potential of a node or edge is high for a given label $(y_i)$ or a label pair $(y_i, y_j)$, the conditional probability of these labels given the features is also high. The conditional probability that is represented by the network can be expressed as

$$P(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{N} \varphi(\mathbf{x}_i, y_i) \prod_{(ij) \in E} \psi(\mathbf{x}_{ij}, y_i, y_j). \tag{1}$$

Here, $Z$ denotes the *partition function* which by definition is given as $Z = \sum_{\mathbf{y}'} \prod_{i=1}^{N} \varphi(\mathbf{x}_i, y_i') \prod_{(ij) \in E} \psi(\mathbf{x}_{ij}, y_i', y_j')$.

It remains to describe the node and edge potentials $\varphi$ and $\psi$. In Taskar *et al.*[2004], the potentials are defined using the *log-linear* model. In this model, a weight vector $\mathbf{w}^k$ is introduced for each class label $k = 1, \ldots, K$. The node potential $\varphi$ is then defined so that $\log \varphi(\mathbf{x}_i, y_i) = \mathbf{w}_n^k \cdot \mathbf{x}_i$ where $k = y_i$. Accordingly, the edge potentials are defined as $\log \psi(\mathbf{x}_{ij}, y_i, y_j) = \mathbf{w}_e^{k,l} \cdot \mathbf{x}_i$ where $k = y_i$ and $l = y_j$. Note that there are different weight vectors $\mathbf{w}_n^k \in \mathbb{R}^{d_n}$ and $\mathbf{w}_e^{k,l} \in \mathbb{R}^{d_e}$ for the nodes and edges.

For the purpose of convenience we use a slightly different notation for the potentials, namely

$$\log \varphi(\mathbf{x}_i, y_i) = \sum_{k=1}^{K} (\mathbf{w}_n^k \cdot \mathbf{x}_i) y_i^k \tag{2}$$

$$\log \psi(\mathbf{x}_{ij}, y_i, y_j) = \sum_{k=1}^{K} \sum_{l=1}^{K} (\mathbf{w}_e^{k,l} \cdot \mathbf{x}_{ij}) y_i^k y_j^l, \tag{3}$$

where $y_i^k$ is an indicator variable which is 1 if point $\mathbf{p}_i$ has label $k$ and 0, otherwise.

In a further refinement step of our model, we introduce the constraints $\mathbf{w}_e^{k,l} = \mathbf{0}$ for $k \neq l$ and $\mathbf{w}_e^{k,k} \geq \mathbf{0}$. This results in $\psi(\mathbf{x}_{ij}, k, l) = 1$ for $k \neq l$ and $\psi(\mathbf{x}_{ij}, k, k) = \lambda_{ij}^k$, where $\lambda_{ij}^k \geq 1$. The idea here is that edges between nodes with different labels should be penalized over edges between equally labeled nodes. This last specification of AMNs makes it possible to run the inference step efficiently using *graph cuts* (see [Boykov *et al.*, 1999; Taskar, 2004]).

## 4 Learning and Inference with AMNs

In this section, we describe how learning and inference can be carried out with AMNs. First, we reformulate Equation (1) as the conditional probability $P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x})$ where the parameters $\omega$ are expressed by the weight vectors $\mathbf{w} = (\mathbf{w}_n, \mathbf{w}_e)$. By plugging in Equations (2) and (3) we obtain that $\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x})$ equals

$$\sum_{i=1}^{N}\sum_{k=1}^{K}(\mathbf{w}_n^k \cdot \mathbf{x}_i)y_i^k + \sum_{(ij)\in E}\sum_{k=1}^{K}(\mathbf{w}_e^{k,k} \cdot \mathbf{x}_{ij})y_i^k y_j^k - \log Z_{\mathbf{w}}(\mathbf{x}). \quad (4)$$

Note that the partition function $Z$ only depends on $\mathbf{w}$ and $\mathbf{x}$, but not on the labels $\mathbf{y}$.

### 4.1 Learning

As mentioned above, in a standard supervised learning task the goal is to maximize $P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x})$. However, the problem that arises here is that the partition function $Z$ depends on the weights $\mathbf{w}$. This means that when maximizing $\log P_{\mathbf{w}}(\hat{\mathbf{y}} \mid \mathbf{x})$, the intractable calculation of $Z$ needs to be done for each $\mathbf{w}$. However, if we instead maximize the *margin* between the optimal labeling $\hat{\mathbf{y}}$ and any other labeling $\mathbf{y}$ defined by

$$\log P_{\omega}(\hat{\mathbf{y}} \mid \mathbf{x}) - \log P_{\omega}(\mathbf{y} \mid \mathbf{x}), \quad (5)$$

the term $Z_{\mathbf{w}}(\mathbf{x})$ cancels out and the maximization can be done efficiently. This method is referred to as *maximum margin* optimization. The details of this formulation are omitted here for the sake of brevity. We only note that the problem is reduced to a quadratic program (QP) of the form:

$$\begin{aligned} &\min \frac{1}{2}\|\mathbf{w}\|^2 + c\xi \quad (6)\\ &\text{s.t. } \mathbf{w}\mathbf{X}\hat{\mathbf{y}} + \xi - \sum_{i=1}^{N}\alpha_i \geq N; \quad \mathbf{w}_e \geq 0;\\ &\alpha_i - \sum_{ij,ji\in E}\alpha_{ij}^k - \mathbf{w}_n^k \cdot \mathbf{x}_i \geq -\hat{y}_i^k, \quad \forall i,k;\\ &\alpha_{ij}^k + \alpha_{ji}^k - \mathbf{w}_e^k \cdot \mathbf{x}_{ij} \geq 0, \quad \alpha_{ij}^k, \alpha_{ji}^k \geq 0, \quad \forall ij \in E,k \end{aligned}$$

Here, the variables that are solved for in the QP are the weights $\mathbf{w} = (\mathbf{w}_n, \mathbf{w}_e)$, a slack variable $\xi$ and additional variables $\alpha_i$, $\alpha_{ij}$ and $\alpha_{ji}$. Again, we refer to Taskar *et al.*[2004] for details.

### 4.2 Inference

Once the optimal weights $\mathbf{w}$ have been calculated, we can perform inference on an unlabeled test data set. This is done by finding the labels $\mathbf{y}$ that maximize $\log P_{\mathbf{w}}(\mathbf{y} \mid \mathbf{x})$. As mentioned above, $Z$ does not depend on $\mathbf{y}$ so that the maximization in equation (4) can be carried out without considering the last term. With the constraints imposed on the variables $y_i^k$ this leads to a linear program of the form

$$\begin{aligned} &\max \sum_{i=1}^{N}\sum_{k=1}^{K}(\mathbf{w}_n^k \cdot \mathbf{x}_i)y_i^k + \sum_{ij\in E}\sum_{k=1}^{K}(\mathbf{w}_e^k \cdot \mathbf{x}_{ij})y_{ij}^k \quad (7)\\ &\text{s.t. } y_i^k \geq 0, \quad \forall i,k; \quad \sum_{k=1}^{K} y_i = 1, \quad \forall i\\ &y_{ij}^k \leq y_i^k, \quad y_{ij}^k \leq y_j^k, \quad \forall ij \in E,k \end{aligned}$$

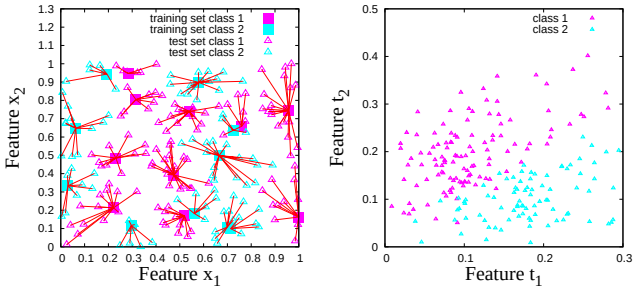



Figure 1: Example of the feature transform $\tau$ for a two-class problem with two features. **Left:** Training data and test data with ground truth labeling. **Right:** Test data after applying $\tau$.

Here, we introduced variables $y_{ij}^k$ representing the labels of two points connected by an edge. The last two inequality conditions are a linearization of the constraint $y_{ij}^k = y_i^k \wedge y_j^k$.

In our current implementation, we perform the learning and inference step by solving the quadratic and linear program from Equations (6) and (7). The implementation is based on the C++ library OOQP [Gertz and Wright, 2003]. As mentioned above, the inference step can be performed more efficiently using graph-cuts. However, in our application, in which the data sets were comparably small, the linear program solver turned out to be fast enough.

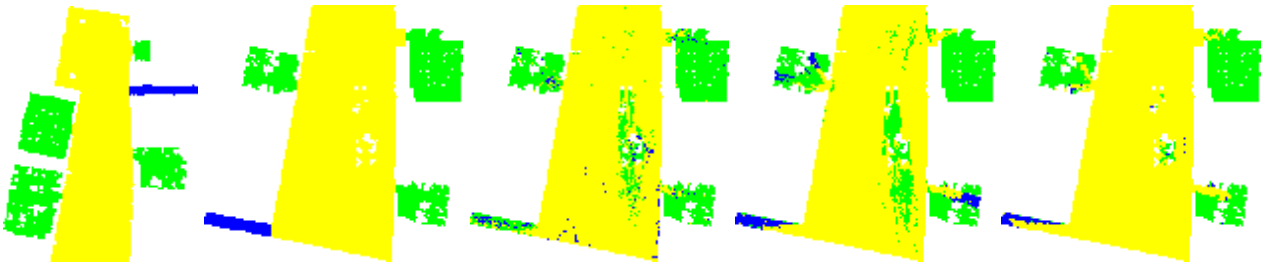
## 5 Instance-based Extension

The main drawback of the AMN classifier, which is based on the log-linear model, is that it separates the classes linearly. This assumes that the features are separable by hyperplanes, which is not justified in all applications. This does not hold for *instance-based* classifiers such as the nearest-neighbor classifier (NN). In NN classification, a query data point $\tilde{\mathbf{p}}$ is assigned to the label that corresponds to the training data point $\mathbf{p}$ whose features $\mathbf{x}$ are closest to the features $\tilde{\mathbf{x}}$ of $\tilde{\mathbf{p}}$. In the learning step, NN simply stores the entire training data set and does not compute a reduced set of training parameters, which is why it is also called *lazy classification*.

The idea now is to combine the advantage of instance-based NN classification with the AMN approach to obtain a collective classifier that is not restricted to the linear separability requirement. This will be presented in the next section.

### 5.1 The Transformed Feature Vector

Suppose we are given a general $K$-class classification problem where for each training data point $\hat{\mathbf{p}}$ we are given a feature vector $\hat{\mathbf{x}}$ of length $L$ and a label $\hat{y} \in \{1,\ldots,K\}$. Now, if we assume that the correct label for a new query feature vector $\tilde{\mathbf{x}}$ corresponds to the label $\hat{y}^*$ of its closest training example $\mathbf{x}^*$ in feature space, then the NN algorithm yields the optimal classification. Of course, this assumption is not valid in general, but we can at least say that the label $\hat{y}^*$ will be more *likely* than any other label. As an example, consider the two-class problem with two features depicted in the left part of Fig. 1. The two-dimensional feature space is defined by the training data shown as boxes. Now, the true labels of an arbitrary test data set, here shown as triangles, will be related



(a) Training data (b) Test data (ground truth) (c) NN (d) AMN (e) inst. AMN

Figure 2: Results on 2D data of an occupancy grid map.

to the labels of their closest training feature points. In our example, the probability of the label $\tilde{y} \in \{1, 2\}$ corresponding to $\tilde{\mathbf{x}}$ is proportional to a Gaussian distribution $\mathcal{N}(\mu_k, \sigma)$ with $\mu_k = d(\tilde{\mathbf{x}}, \hat{\mathbf{x}}_k)$ and $k \in \{1, 2\}$. Here, $\hat{\mathbf{x}}_k$ denotes the training example with label $k$ that is closest to $\tilde{\mathbf{x}}$ and $d(\cdot, \cdot)$ the distance in feature space. The variance $\sigma$ is set to 0.03.

From the left side of Fig. 1, we can see that any attempt to separate the classes using hyperplanes (in this case lines) will lead to severe classification errors. However, if we introduce a *feature transform* $\tau : \mathbb{R}^L \to \mathbb{R}^K$ in such a way that $\tau(\tilde{\mathbf{x}}) = (d(\tilde{\mathbf{x}}, \hat{\mathbf{x}}_1), \ldots, d(\tilde{\mathbf{x}}, \hat{\mathbf{x}}_K))$, then the transformed features $\tilde{\mathbf{t}} := \tau(\tilde{\mathbf{x}})$ are more easily separable by hyperplanes. This is illustrated on the right side of Fig. 1. The reason for this is that the NN classifier always chooses the label corresponding to the smallest component of $\tilde{\mathbf{t}}$. This means, that the set $\mathcal{T}_k$ of all transformed feature points $\mathbf{t} = (t_1, \ldots, t_K)$ that will be assigned the label $k$ can be described by

$$\mathcal{T}_k = \{\mathbf{t} \in \mathbb{R}^K | t_k < t_1 \wedge \cdots \wedge t_k < t_{k-1} \wedge t_k < t_{k+1} \wedge \cdots \wedge t_k < t_K\}.$$

As can be seen from this equation, the border of the set $\mathcal{T}_k$ is described by $K - 1$ hyperplanes passing through the origin. In the case of our two-class example, the only separating hyperplane (line) is described by the equation $t_1 = t_2$.

### 5.2 $M$ Nearest Neighbors

One problem of the NN classifier is that the assignment of a label to a query point $\tilde{\mathbf{p}}$ only depends on the labeling of *one* instance in the training set, namely the one whose feature vector is closest to $\tilde{\mathbf{x}}$. However, it is possible that the features of other training instances are also very close to $\tilde{\mathbf{x}}$, although they are labeled differently. For example, suppose that the distances of $\tilde{\mathbf{x}}$ to the two closest training features $\hat{\mathbf{x}}^1$ and $\hat{\mathbf{x}}^2$ are very similar, and the corresponding labels $\hat{y}^1$ and $\hat{y}^2$ are different, the decision of assigning the label $\hat{y}^1$ to $\tilde{\mathbf{p}}$ may be wrong, especially in the presence of noise. Therefore we proceed as follows: For the feature vector $\tilde{\mathbf{x}}$ that corresponds to $\tilde{\mathbf{p}}$ we compute the $M$ nearest training instances in each of the $K$ classes $C_1, \ldots, C_K$ and the corresponding distances $d(\tilde{\mathbf{x}}, \hat{\mathbf{x}}_k^m)$ where $k = 1, \ldots, K$ and $m = 1, \ldots, M$. These are used to define the transformed feature vector $\tau_M(\tilde{\mathbf{x}})$ as

$$\tau_M(\tilde{\mathbf{x}}) = (\ldots, d(\tilde{\mathbf{x}}, \hat{\mathbf{x}}_k^1), \ldots, d(\tilde{\mathbf{x}}, \hat{\mathbf{x}}_k^M), \ldots) \tag{8}$$

Experiments show that higher values of $M$ increase the classification rate, but for large $M$ the improvement is very small. In our experiments, $M = 5$ turned out to be a good choice.

## 6 Experimental Results

We performed a series of experiments with 2D and 3D data to compare our *instance–based AMN* (inst. AMN) algorithm, with the NN and the AMN classifier. The results of these experiments demonstrate that our inst. AMN outperforms the two other algorithms, independent of the features used.

### 6.1 Implementation Details

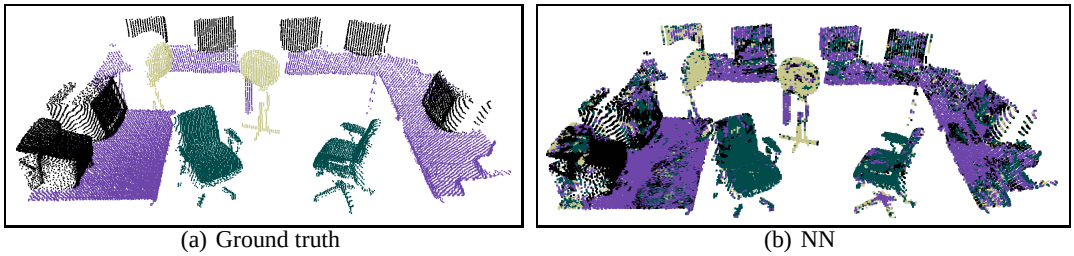
To speed up the learning process, we need to represent all feature vectors such that the nearest neighbor search in the feature space can be performed efficiently. To this end, we use $kD$-trees $\mathcal{K}_1, \ldots, \mathcal{K}_K$ to store the training feature vectors of each class $C_1, \ldots, C_K$. This way, the computational effort of the nearest neighbor lookup becomes logarithmic in the number of the stored instances.

Thus, the training step consists of computing the features for the training data, transforming these features according to Eq. 8 and assigning the transformed features to the nodes of the AMN. The edge features of the AMN consist of a constant scalar value, as described by Anguelov *et al.* [2005]. After solving the quadratic program we obtain the weight vectors $\mathbf{w}^k$. Then, in the inference step, we use the transformed features $\tau_M(\mathbf{x})$ of the test data as node features for the AMN. Again, the edge features are constant.

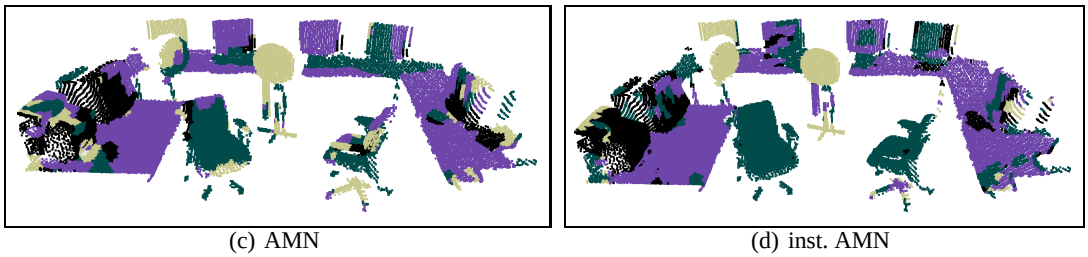
**Feature Computation** Depending on the input data used, we computed different types of features for the data points. In the case of the 2D grid data, each point in the map is represented by a set of geometrical features which are extracted from a laser range scan covering 360° field of view. Each laser is simulated and centered at each point in the map representing a free space [Mozos *et al.*, 2005]. Because the number of geometrical features is big (more than 300) we select the best ones using the AdaBoost algorithm.

For the 3D data set we computed *spin images* [Johnson, 1997] with a size of $5 \times 10$ bins. The spherical neighborhood for computing the spin images had a radius between 10 and 15$cm$, depending on the resolution of the input data.

**Data Reduction** When classifying data sets with many points, e.g., 3D scans with over $10,000$ scan points, the time and memory requirements of the AMN classifier grows very large. In these cases, we perform a data set reduction, again using $kD$-trees: after inserting all data points into the tree, we



(a) Ground truth (b) NN



(c) AMN (d) inst. AMN

Figure 3: Classification results for one scene from the MULTI data set

prune the tree at a fixed level $\lambda$. All points in the subtrees below level $\lambda$ are then merged into one mean point. The spin image features, however, are still computed on the whole data set to provide a high density in the feature extraction.

### 6.2 2D Map Annotation

For the 2D classification experiment we used an occupancy grid map of the interior of a building. The map was annotated with three different labels, namely 'corridor', 'room' and 'lobby'. Then the map was divided into two non-overlapping submaps, one for training and one for testing. Fig. 2(a) shows the submap used for training and Fig. 2(b) the ground truth for the classification. The results obtained with the NN and the standard AMN approach are shown in Figs. 2(c) and 2(d), while Fig 2(e) shows the result of our inst. AMN algorithm. It can be seen that the inst. AMN approach gives the best result. The classification rate of the NN is with 92.2% higher than that of the standard AMN, which was 89.8%, but the classification is very noisy. In contrast, the inst. AMN result is more consistent wrt. neighboring data points and has the highest classification rate with 95.5%.

### 6.3 3D Scan Point Classification

Furthermore, we evaluated the classification algorithms on three different 3D data sets with an overall number of 38 scanned scenes. The scans were obtained with a 3D laser range scanner. The first data set is called HUMAN and consists of 11 recorded scenes with two humans in varying poses. The scans were labeled into the four classes 'head', 'torso', 'legs', and 'arms'. The second data set named SINGLE consists of 20 different 3D scans of the object classes 'chair', 'table', 'screen', 'fan', and 'trash can'. Each scan in the SINGLE data set contains just one object of each class, apart from tables, which may have screens standing on top of them. The last data set is named MULTI and consists of seven scans with multiple objects of the same types as in the SINGLE data set.

The HUMAN and SINGLE data sets were evaluated using cross validation. For the MULTI data set we used the object instances from the SINGLE set for training, because those were not occluded by other objects. The classification was performed on the complete scans.

Fig. 3 shows a typical classification result for a scan from the MULTI test set. We can see that NN assigns wrong labels to different points while their neighbors are classified correctly. The AMN results show that areas of points tend to be classified with the same label. However, due to the restriction to linear separable data, many object parts are misclassified, especially in complex objects like chairs and fans. In contrast, the results obtained with inst. AMN are better even in these complex objects, because the transformed feature vectors computed by inst. AMN are better suited for a classification based on separating hyper-planes. Table 1 shows the resulting classification rates. We can see that the inst. AMN classifier outperforms both of the others in all three data sets.

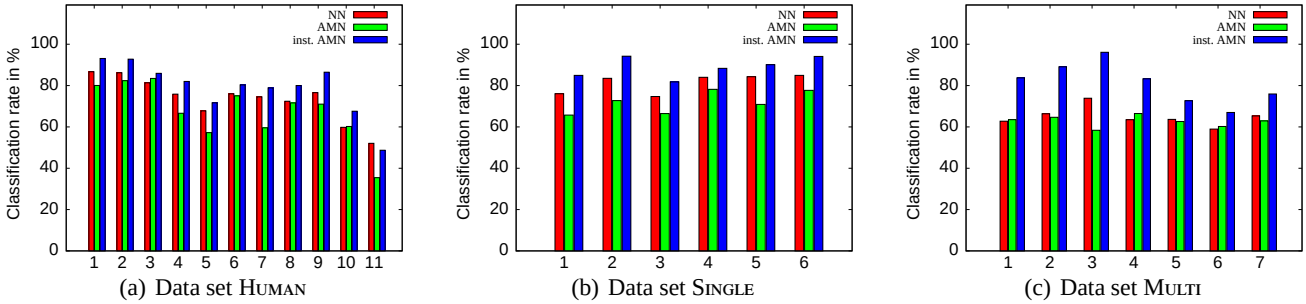
A statistical analysis using a *t*-student test with a significance level of 0.05 is shown in Fig. 5. As can be seen, for the MULTI set our algorithm performs significantly better than both the inst. AMN and standard AMN. A detailed analysis of the classification results is depicted in Fig. 4 wchich demonstrates that our inst. AMN yields highly accurate results for all three data sets.

| Data set | NN | AMN | inst. AMN |
|---|---|---|---|
| HUMAN | 75% | 69% | 80% |
| SINGLE | 81% | 72% | 89% |
| MULTI | 63% | 62% | 76% |

Table 1: Classification results for all three data sets

## 7 Conclusions

In this paper we proposed an algorithm that combines associative Markov networks with an instance-based approach similar to the nearest neighbor classifier for identifying ob-




(a) Data set HUMAN (b) Data set SINGLE (c) Data set MULTI

Figure 4: Individual classification rates for all scenes.

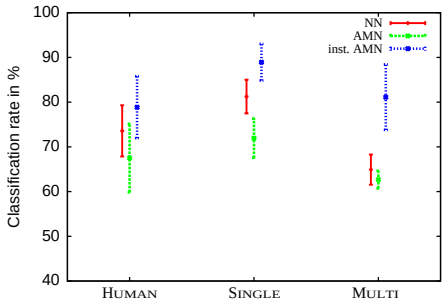



Figure 5: Results of a *t*-student test with significance level 0.05 on the three different 3D data sets. For the MULTI data set we obtain a significant improvement over NN and standard AMN classification.

jects in 3D range data. In contrast to the approach suggested by Anguelov *et al.* [2005], our method is able to classify more complex objects with a diverse set of features per class. By using the distances of features to their nearest neighbors, the transformed feature space becomes linearly separable. Accordingly, it improves the performance of the AMN training step. While the AMN inference task can be solved using graph-cuts, the drawback of our approach is, that by storing instances the resulting classifier becomes a lazy classification method. The inference step requires to compute the distance between the instance to be classified and the known training instances. This is computational expensive in general. However, by utilizing efficient data structures like *k*D-Trees the computational effort becomes logarithmic in the number of stored instances.

Despite these encouraging results, there are several aspects that warrant future research. One way to improve the approach presented here could be to classify the objects on a lower complexity level, i.e., by classifying object parts following the idea by Huber *et al.* [2004]. As features of less complex objects are distributed more compactly in the feature space, they might be easier to separate.

## Acknowledgments

This work has partly been supported by the German Research Foundation under contract number SFB/TR8 and by the European Union under contract number FP6-004250-CoSy.

## References

[Anguelov *et al.*, 2005] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *Proc. of the Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 169–176, 2005.

[Boykov and Huttenlocher, 1999] Y. Boykov and D. Huttenlocher. A new Bayesian approach to object recognition. In *Proc. of IEEE CVPR*, 1999.

[Boykov *et al.*, 1999] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *Proc. of the Int. Conf. on Computer Vision (ICCV)*, pages 377–384, 1999.

[Chakrabarti and Indyk, 1998] S. Chakrabarti and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proc. of the ACM SIGMOD*, 1998.

[de Alarcón *et al.*, 2002] P. A. de Alarcón, A. D. Pascual-Montano, and J. M. Carazo. Spin images and neural networks for efficient content-based retrieval in 3d object databases. In *Proc. of the Int. Conf. on Image and Video Retrieval (CIVR)*, 2002.

[Frome *et al.*, 2004] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik. Recognizing objects in range data using regional point descriptors. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2004.

[Gertz and Wright, 2003] E. M. Gertz and S. J. Wright. Object-oriented software for quadratic programming. *ACM Trans. on Mathematical Software*, pages 58–81, 2003.

[Huber *et al.*, 2004] D. Huber, A. Kapuria, R. Rao Donamukkala, and M. Hebert. Parts-based 3d object classification. In *Proc. of the Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[Johnson, 1997] A. Johnson. *Spin-Images: A Representation for 3-D Surface Matching*. PhD thesis, Robotics Institute, Carnegie Mellon Univ., Pittsburgh, PA, 1997.

[Li and Guskov, 2005] X. Li and I. Guskov. Multiscale features for approximate alignment of point-based surfaces. In *Symp. on Geometry Processing*, pages 217–226, 2005.

[Limketkai *et al.*, 2005] B. Limketkai, L. Liao, and D. Fox. Relational object maps for mobile robots. In *Proc. of the Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1471–1476, 2005.

[Mian *et al.*, 2004] Ajmal S. Mian, Mohammed Bennamoun, and Robyn A. Owens. Matching tensors for automatic correspondence and registration. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2004.

[Mozos *et al.*, 2005] O. Martinez Mozos, C. Stachniss, and W. Burgard. Supervised learning of places from range data using adaboost. In *Proc. of the Int. Conf. on Robotics & Automation (ICRA)*, 2005.

[Osada *et al.*, 2001] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin. Matching 3D models with shape distributions. In *Shape Modeling International*, pages 154–166, 2001.

[Ruiz-Correa *et al.*, 2003] S. Ruiz-Correa, L. G. Shapiro, and M. Meila. A new paradigm for recognizing 3-d object shapes from range data. In *Proc. of the Int. Conf. on Computer Vision (ICCV)*, pages 1126–1133, 2003.

[Taskar *et al.*, 2002] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Eighteenth Conf. on Uncertainty in Artificial Intelligence (UAI02)*, 2002.

[Taskar *et al.*, 2004] B. Taskar, V. Chatalbashev, and D. Koller. Learning associative markov networks. In *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2004.

[Taskar, 2004] Ben Taskar. *Learning Structured Prediction Models: A Large Margin Approach*. PhD thesis, Stanford University, Stanford, CA, December 2004.

[Wu *et al.*, 2004] Z. Wu, Y. Wang, and G. Pan. 3D face recognition using local shape map. In *Proc. of IEEE Intern. Conf. on Image Processing*, pages 2003–2006, 2004.