Automatic Selection of Object Recognition Methods using Reinforcement Learning

Reinaldo A. C. Bianchi, Arnau Ramisa and Ramón López de Mántaras

Abstract Selecting which algorithms should be used by a mobile robot computer vision system is a decision that is usually made *a priori* by the system developer, based on past experience and intuition, not systematically taking into account information that can be found in the images and in the visual process itself to learn which algorithm should be used, in execution time. This paper presents a method that uses Reinforcement Learning to decide which algorithm should be used to recognize objects seen by a mobile robot in an indoor environment, based on simple attributes extracted on-line from the images, such as mean intensity and intensity deviation. Two state-of-the-art object recognition algorithms can be selected: the constellation method proposed by Lowe together with its interest point detector and descriptor, the Scale-Invariant Feature Transform and Nistér and Stewénius Vocabulary Tree approach. A set of empirical evaluations was conducted using a image database acquired with a mobile robot in an indoor environment, and results obtained shows that the approach adopted here is very promising.

1 Introduction

Object recognition is a central theme in Computer Vision, with a large variety of applications ranging from medical image analysis to industrial inspection. Unfortunately, there is also a large variety of solutions to the object recognition problem, producing a wider still range of results, depending on the domain of the problem, the sensor hardware, the lightning conditions and the object to be identified. An autonomous mobile robot may employ an object recognition system for a variety of

Reinaldo A. C. Bianchi
Centro Universitário da FEI, São Bernardo do Campo, Brazil, e-mail: rbianchi@fei.edu.br

Arnau Ramisa and Ramón López de Mántaras
Artificial Intelligence Research Institute (IIIA-CSIC), Bellaterra, Spain. e-mail: {aramisa, mantaras}@iiia.csic.es

tasks, such as navigation, localization and mapping or interacting with humans. For example, precisely identifying and locating objects in the images means being able to localize themselves better in the real world.

When one thinks about designing an autonomous mobile robot capable of operating in the real world, and the variety of conditions that it will face, one can conclude that choosing, a priori, which object recognition method a robot should have, is not the best design option. In this kind of application, the robot should be able to decide by itself which object recognition method should be used, depending on the current conditions of the world.

In this paper we propose the use of Reinforcement Learning to design a mobile robot that is capable of deciding, during on line world exploration, which method should be used to identify objects in an indoor environment, aiming also to minimize computing time. To evaluate this idea we implemented a system that is able to choose between two well known object recognition algorithms based on simple attributes extracted on-line from the images, such as mean intensity and intensity deviation. In addition, it is also capable of deciding that an image is not suitable for analysis, and thus discard it.

This paper is organized as follows: Section 2 briefly reviews Reinforcement Learning and its use as a technique to optimize computer vision, image segmentation and object recognition algorithms. Section 3 describes the two object recognition methods used by our robot. Section 4 describe how to use RL to automatic select the object recognition method to be used by the robot and Section 5 describes the domain where this proposal has been evaluated and the results obtained. Finally, Section 6 summarizes some important points learned from this research and outlines future work.

2 Reinforcement Learning and its applications in Computer Vision

Reinforcement Learning (Sutton and Barto, 1998) is concerned with the problem of learning from interaction to achieve a goal, for example, an autonomous agent interacting with its environment via perception and action. On each interaction step the agent senses the current state s of the environment, and chooses an action a to perform. The action a alters the state s of the environment, and a scalar reinforcement signal r (a reward or penalty) is provided to the agent to indicate the desirability of the resulting state. In this way, "The RL problem is meant to be a straightforward framing of the problem of learning from interaction to achieve a goal" (Sutton and Barto, 1998).

Formally, the RL problem can be formulated as a discrete time, finite state, finite action Markov Decision Process (MDP) (Mitchell, 1997). Given:

- finite set of states $s \in \mathscr{S}$ that the agent can achieve;
- A finite set of possible actions $a \in \mathscr{A}$ that the agent can perform;

- A state transition function $\mathscr{T} : \mathscr{S} \times \mathscr{A} \rightarrow \Pi(\mathscr{S})$, where $\Pi(\mathscr{S})$ is a probability distribution over $\mathscr{S}$;
- A finite set of bounded reinforcements (payoffs) $\mathscr{R} : \mathscr{S} \times \mathscr{A} \rightarrow \Re$,

the task of a RL agent is to find out a stationary policy of actions $\pi^* : \mathscr{S} \rightarrow \mathscr{A}$ that maps the current state s into an optimal action(s) a to be performed in s, maximizing the expected long term sum of values of the reinforcement signal, from any starting state.

The policy π is some function that tells the agent which actions should be chosen, and is learned through trial-and-error interactions of the agent with its environment. Several algorithms were proposed as a strategy to learn an optimal policy π^* when the model ($\mathscr{T}$ and $\mathscr{R}$) is not known in advance, for example, the Q–learning (Watkins, 1989) and the SARSA (Rummery and Niranjan, 1994) algorithms.

The Q–learning algorithm was proposed by Watkins (1989) as a strategy to learn an optimal policy π^* when the model ($\mathscr{T}$ and $\mathscr{R}$) is not known in advance. Let $Q^*(s,a)$ be the reward received upon performing action a in state s, plus the discounted value of following the optimal policy thereafter:

$$Q^*(s,a) \equiv R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') V^*(s'). \tag{1}$$

The optimal policy π^* is $\pi^* \equiv \arg\max_a Q^*(s,a)$. Rewriting $Q^*(s,a)$ in a recursive form:

$$Q^*(s,a) \equiv R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \max_{a'} Q^*(s',a'). \tag{2}$$

Let $\hat{Q}$ be the learner's estimate of $Q^*(s,a)$. The Q–learning algorithm iteratively approximates $\hat{Q}$, i.e., the $\hat{Q}$ values will converge with probability 1 to Q^*, provided the system can be modeled as a MDP, the reward function is bounded ($\exists c \in \mathscr{R}; (\forall s,a), |R(s,a)| < c$), and actions are chosen so that every state-action pair is visited an infinite number of times. The Q learning update rule is:

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha \left[r + \gamma \max_{a'} \hat{Q}(s',a') - \hat{Q}(s,a) \right], \tag{3}$$

where s is the current state; a is the action performed in s; r is the reward received; s' is the new state; γ is the discount factor ($0 \leq \gamma < 1$); $\alpha = 1/(1 + visits(s,a))$, where $visits(s,a)$ is the total number of times this state-action pair has been visited up to and including the current iteration.

Several researchers have been using RL as a technique to optimize active vision, image segmentation and object recognition algorithms. The area of Computer Vision on which RL was first applied was in Active Vision. Whitehead and Ballard (1991) proposed an adaptive control architecture to integrate active sensory-motor systems with RL based decision systems. Although the work is theoretical and did not make use of real sensors, they were able to describe a system that learns to focus its attention on the relevant aspects of the domain as well as control its behavior, in a simple block manipulation task. Several researchers have been applying

RL to active vision since then, for example Minut and Mahadevan (2001) have applied RL for visual attention control, proposing a model of selective attention for visual search tasks, such as deciding where to fixate next in order to reach the region where an object is most likely to be found. Darrell and Pentland (1996a,b) also address visual attention problem: they proposed a gesture recognition system that guides an active camera to foveate salient features based on a reinforcement learning paradigm. An attention module selects targets to foveate based on the goal of successful recognition, learning where to foveate to maximally discriminate a particular gesture. Finally, in Darrell (1998) is shown how a concise representation of active recognition behavior can be derived from hidden-state reinforcement learning techniques.

Paletta and Pinz (2000) have applied RL in an active object recognition system, to learn how to move the camera to informative viewpoints, defining the recognition process as a sequential decision problem with the objective of disambiguating initial object hypotheses. For these authors, "Reinforcement Learning provides then an efficient method to autonomously develop near-optimal decision strategies in terms of sensorimotor mappings" (Paletta et al, 1998). Borotschnig et al (1999) continued in the same line of work, building a system that learns to reposition the camera to capture additional views to improve the image classification result obtained from a single view. More recently, Paletta et al (2005) proposed the use of Q-learning to associate shift of attention actions to cumulative reward with respect to object recognition. In this way, the agent learns sequences of shifts of attentions that lead to scan paths that are highly discriminative with respect to object recognition.

Less work have been done on the use of RL for image segmentation and object recognition. Peng and Bhanu (1998a) used RL to learn, from input images, to adapt the image segmentation parameters of a specific algorithm to the changing environmental conditions, in a closed-loop manner. In this case, the RL creates a mapping from input images to corresponding segmentation parameters. This contrasts with great part of the current computer vision systems whose methodology is open-loop, using image segmentation followed by object recognition algorithms. Peng and Bhanu (1998b) improves the recognition results over time by using the output at the highest level as feedback for the learning system, and has been used to learn the parameters of image segmentation and feature extraction and thereby recognizing 2-D objects, systematically controlling feedback in a multilevel vision system. The same authors presented a general approach to image segmentation and object recognition that can adapt the image segmentation algorithm parameters to the changing environmental conditions, in which segmentation parameters are represented by a team of generalized stochastic learning automata and learned using connectionist reinforcement learning techniques. Results were presented for both indoor and outdoor color images, showing a performance improvement over time for both image segmentation and object recognition using RL (Bhanu and Peng, 2000).

Taylor (2004) also followed this line of research, applying RL algorithms to learn parameters of an existing image segmentation algorithm. Using the Fuzzy ARTMAP artificial neural network, he was able to optimize ten parameters of Wolf and Jolion

(2003) algorithm for text detection in still images. The parameters learned by RL were shown to be superior to the parameters previously recommended. Other applications of RL to learn parameters of image segmentation algorithms includes: contrast adaptation (Tizhoosh and Taylor, 2006), finding the appropriate threshold in order to convert an image to a binary one (Yin, 2002; Shokri and Tizhoosh, 2003, 2004, 2008; Sahba et al, 2008) and detection of patterns in satellite images (Hossain et al, 1999).

Finally, Draper et al (1999) modeled the object recognition problem as a Markov Decision Problem, and proposed a theoretically sound method for constructing object recognition strategies by combining CV algorithms to perform segmentation. The authors tested their method in a real system, learning sequences of image processing operators for detecting houses in aerial images.

Reinforcement Learning has been widely used in the Computer Vision field in particular cases, mainly: to optimize the performance of active vision systems; to decide where the focus of attention should be in order to accomplish a certain task; to learn how to move a camera to more informative viewpoints and; to optimize parameters of existing and new computer vision algorithms, such as thresholds, contrast and internal parameters. In these cases, the resulting systems and algorithms have been very successful ones.

RL has also been used for constructing object segmentation and recognition strategies by combining CV algorithms. However, results in this area have not been as good as those of RL applied to active vision or parameter optimization: it usually has been limited to a very specific image domain, such as the one in Draper et al (1999).

The main limitations which arise when using reinforcement learning are, first, that the reward value associated with a situation is usually not directly available, and thus rewards are results of indirect definitions. RL requires that a certain amount of knowledge about the world is available in the form of a training set, which is not the case in many vision tasks. Second, the large space state which difficult convergence of RL algorithms raises performance issues, as the learning phase can take a long time.

3 Two Object Recognition Methods

Two successful general object recognition approaches that have been widely used are the constellation method proposed by Lowe together with its interest point detector and descriptor SIFT (Lowe, 2004) and the Vocabulary Tree algorithm proposed by Nistér and Stewénius (2006).

The first approach is a single view object detection and recognition system which has some interesting characteristics for mobile robots, most significant of which are the ability to detect and recognize objects at the same time in an unsegmented image and the use of an algorithm for approximate fast matching, which reduces the search time by two orders of magnitude for a database of 100,000 keypoints for a 5% loss

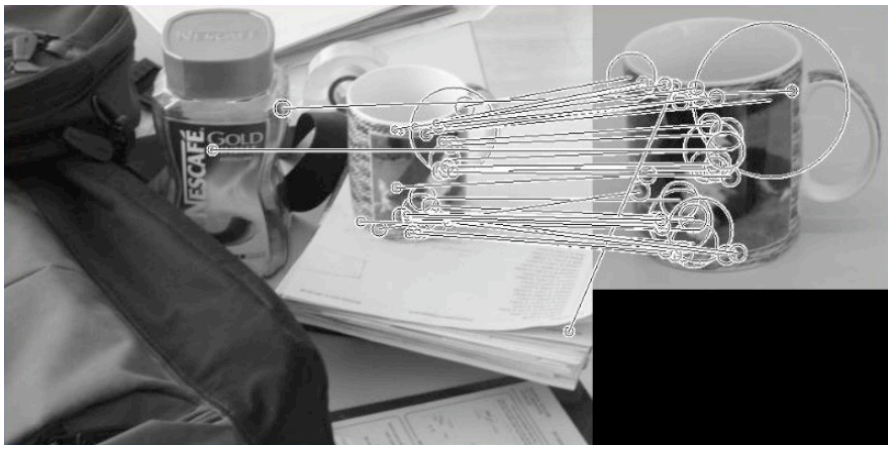

Fig. 1 Matching stage in the Lowe object recognition method.

in the number of correct matches. In this approach, individual descriptors of the features detected in a test image are initially matched to the ones stored in the object database using the Euclidean distance. False matches are rejected if the distance of the first nearest neighbor is not distinctive enough when compared with that of the second. In Figure 1, the matching features between a test and model images can be seen. The presence of some outliers can also be observed.

Once a set of matches is found, the generalized Hough transform is used to cluster each match of every database image depending on its particular transformation (translation, rotation and scale change). Although imprecise, this step generates a number of initial coherent hypotheses and removes a notable portion of the outliers that could potentially confuse more precise but also more sensitive methods. All clusters with at least three matches for a particular image are accepted, and fed to the next stage: a Least Squares is used to improve the estimation of the affine transformation between the model and the test images.

Because of the necessarily broad clusters of the Hough transform, some erroneous matches can still be present and need to be removed. In order to do so, in this work a RANSAC step is added to the Lowe approach. RANSAC labels non-coherent matches as outliers and, additionally, estimates the most probable affine transformation for every hypothesis given its initial set of matches. Hypotheses that lose matches below three are discarded. The hypotheses that remain after the RANSAC step are reasonable outlier-free and a more accurate model fitting algorithm like Iterative Reweighed Least Squares can be used. The diagram of the complete algorithm is presented in Figure 2.

The second approach to object classification that is used in this work is Vocabulary Tree method proposed by Nistér and Stewénius (2006). This approach to object classification comes from the text categorization domain, where the occurrence of certain words in documents is recorded and used to train classifiers that can later recognize the subject of new texts. Recently these approach has been adapted to

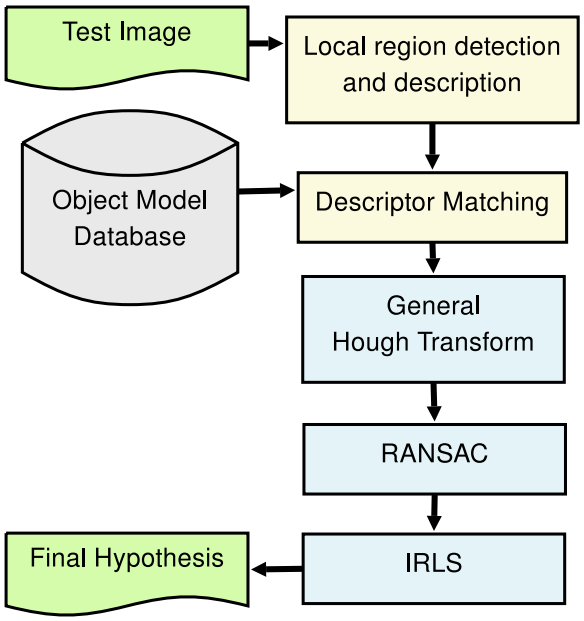

Fig. 2 Diagram for the Lowe algorithm.

visual object recognition by different authors (Sivic et al, 2006; Csurka et al, 2004) using local descriptors computed on image features as visual words.

The visual descriptor space has normally a high dimensionality and, to make it tractable, it is discretized in a codebook. This codebook should be specific enough to distinguish between different descriptor types but general enough to be insensitive to small variations in the local patch. In the Vocabulary Tree approach used in this work, the codebook is a tree built applying hierarchical k-means to a dataset of descriptors. A histogram of descriptor occurrences is built to characterize an image.

Next, a multi-class classifier – the k-NN in this implementation – is trained with the histograms of local descriptor counts. The class of the object in the image is determined as the dominant one in the k nearest neighbors. The diagram of the complete algorithm is presented in Figure 3.

One of the drawbacks of this method is that, in contrast to Lowe's method, is designed to work with pre- segmented images. A straightforward solution to the image segmentation problem is to first apply bilateral filtering to remove texture from the image (Tomasi and Manduchi, 1998; Paris et al, 2007). Next, the Canny edge detector (Canny, 1986) is applied to define the edges in the image and then mathematical morphology operators are applied (one opening followed by and one closing of the image) in order to close the contours that remained open (Haralick et al, 1987). Then, a flood-fill algorithm is used to fill connected areas divided by the edges. A threshold is set to the flood-fill algorithm to preserve strong gradient

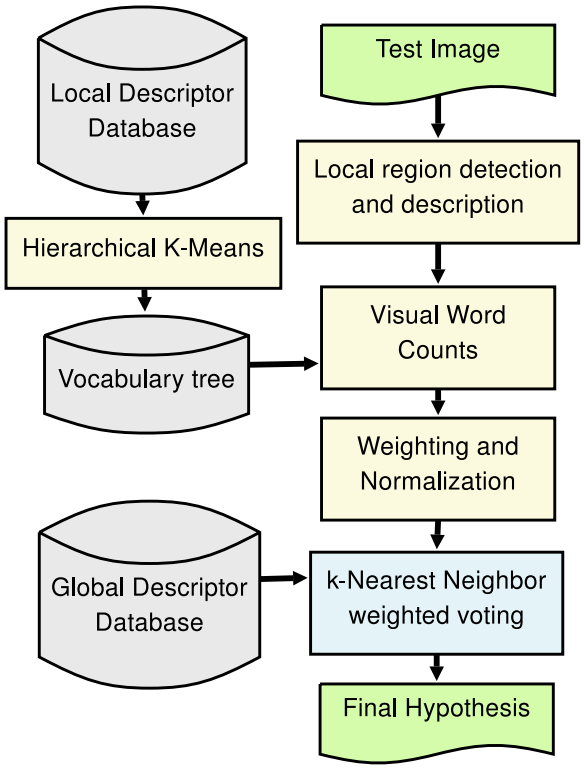

Fig. 3 The Vocabulary Tree algorithm.

jumps that have not been detected by the Canny edge detector. Finally, areas which are too small, too large or to thin are removed from the list of segmented regions.

Although both object recognition methods proved their reliability in real world applications (for example, see Ramisa et al (2008), they have their limitations: Lowe's method performs poorly when recognizing sparsely textured objects or objects with repetitive patterns, while the Vocabulary Tree needs an accurate segmentation stage, prior to classification, which can be very time consuming. Furthermore, the method depends on the quality of that segmentation stage to provide good results.

4 Learning to Select Object Recognition Methods

In order to decide which algorithm should be used by the agent, the RL problem was defined as a 2 stage decision problem, with 2 possible actions in each stage: In the first one, the agent must decide if the image contains an object, and thus must be recognized, or if the image does not contain objects, and can be discarded, saving

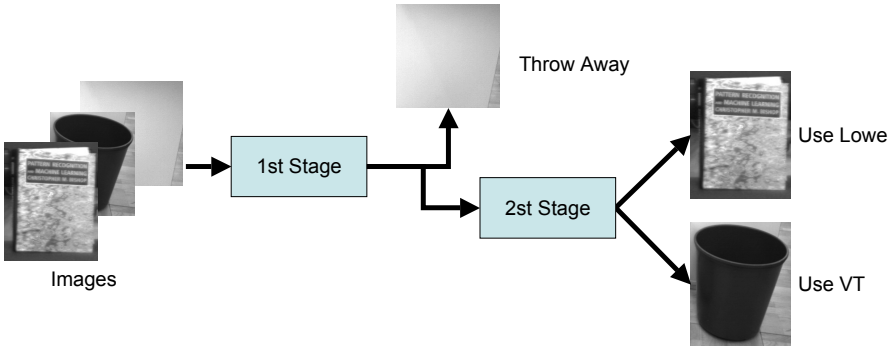

Fig. 4 The two stage decision problem.

processing time. In the second stage, the agent must decide which object recognition algorithms should be used: Lowe's or the Vocabulary Tree (VT) algorithm (see Figure 4).

To learn how to select the object recognition method appropriate for one image at one stage we propose to use Reinforcement Learning as a classification method. In this approach the space state is defined as a combination attributes extracted from the images plus the possible classification of the image. For example, for the first stage, the state can be defined as a combination of mean image intensity and standard deviation and a value defining if the image is a background and can be discarded or if contain objects.

We also define a new type of action, called "update action". Update actions are not real actions happening in the world, but actions that update the value of a state-action pair $Q(s,a)$ at one state using the value of a neighbor pair. For example, if the space state is composed of image intensity and standard deviation, the $Q(s,a)$ table would be represented as two dimensional matrix containing the possible values of intensity (0 to 255) and standard deviation (0 to 255), and update actions are done between one state and his 4-neighbours located above, below, at left and at right. In a 3-dimensional space state, update actions can be made using 8-neighbours (two in each direction of the space) and so on.

The rewards used during the learning phase are computed using a set of training images. If, during the exploration, the learning agent reaches a state where a training image exists and the state corresponds to the correct classification of the image, the agent receives a reward. Otherwise the reward is zero. For example, in the first stage of the decision, if we have a training image that does not contain an object, with mean intensity value of 50 and standard deviation of 10, a reward is given when the agent moves to the state (mean = 50, std = 10, classification = discard). An interesting propriety of this approach is that the rewards can be pre-computed, creating a reinforcement table that can be used during the learning phase (an example is shown in the next section).

Formally, a MDP can be defined for each stage as:

- The set of update actions $a \in \mathscr{A}$ that the agent can perform, defined as update the Q value using the value of a neighbor.
- The finite set of states $s \in \mathscr{S}$ in this case is the n-dimensional space of values of the attributes extracted from the images plus its classification;
- The state transition function allows updates to be made between any pair of neighbors in the set of states.
- The reinforcements $\mathscr{R} : \mathscr{S} \times \mathscr{A} \rightarrow \Re$ are defined using a set of training images.

In this approach, a RL method is used as a classifier, and must have two processing phases: the training phase, where reinforcement learning is performed over a set of pre-classified images, i.e., images to which we know what the best algorithm to use is, and the execution phase, where the results from the learning is used to define which algorithms to apply to other images.

During the training phase, learning an optimal policy to solve the MDP means to learn a mapping from images (or more specifically image attributes) to image classes (or algorithms classes). Although several RL algorithms can be used to do this, the RL algorithm used in this implementation is the Q-learning (Watkins, 1989), because it directly approximates the optimal policy independently of the policy being followed (it is an off-policy method), allowing the state and the action to be executed by the agent to be selected randomly. Using the Q-Learning, at each stage the agent chooses a system state s. Then, it selects an update action to be executed, compute the reward and update the value function.

The learning algorithm used is as follows:

```
Initialize Q(s,a).
Choose a start state s, randomly.
do {
   Choose an action, randomly.
   Execute a, observe s', compute the reward.
   Update the Q value.
   s = s'.
} until the Q values converge.
```

To better understand what is happening during the learning phase, we can compare our approach to a robot moving in a two-dimensional grid. Every time the robot finds a "goal" state and receives a reward, the state-action pair where the robot was before reaching the goal state is updated. Every time the robot moves, it iteratively updates the origin state-action pair. By doing this a large number of times, the reward is spread over the Q-table, and a robot will know what to do to reach the goal state (will have learned the optimal policy).

In the learning phase, every time the "robot" reaches a state where there is an image from the training set, it receives a reward, and the state-action pair where the "robot" was before is updated. Every time a new state action pair is randomly chosen, it is iteratively updated. By doing this a large number of times, the reward is spread over the Q-table, and every state-action pair will contain information about what to do with an image with those characteristics (will have learned a mapping from image to actions).

0000000000	1111111110
0000010000	1111111111
2000011000	2011111111
0000001100	2221111111
0200101000	2222111111
0020200110	2222211111
0000010000	2222111111
0200011000	2220111111
2021000000	2221111111

Fig. 5 Example of how the classification works: the reinforcements used (left) and the resulting classification table (right).

The figure 5 shows an example of the use of RL as an image classification method: in each figure the lines and columns represents two values of the attributes extracted from the images (for example, mean image intensity are lines and standard deviation of the image intensity are the columns), the value represents what is the classification of an image, where zero means that there is not an example with that characteristics, and "1" and "2" are two possible classes.

The figure on the left presents the reinforcement table, created before the training phase using the training images and their classification, which consists of what is the algorithm that was able to classify them. The image on the right shows the results of applying the RL algorithm during the learning phase: a table where the classification was spread over to states where there are no prior examples, and that allows the classification of other images.

To show the applicability of this porposal, experiments and results obtained with this method are presented in the next section.

5 Experiments and Results

To test the algorithm selection method proposed in this work, an image dataset of nine typical household objects (Ramisa et al, 2008) was used. In this dataset, objects are divided in three categories (three objects per category): textured, untextured and textured but with repetitive patterns. Every category consists of three different objects and each object has approximately 20 training images (used for the Vocabulary Tree algorithm). The objects are mugs, books, trashcans, chairs and computer monitors. In Figure 5 one object from each category can be seen.

Also, this image dataset contains test images where the same objects appear. On line, embedded processing means that images acquired by a mobile robot hardly have a resolution greater than one megapixel, and the object to be detected will prob-

Fig. 6 Objects from the dataset. First column corresponds to objects with repetitive texture, second to textured objects and third to non-textured objects.

ably only occupy the image partially. Additionally, movement often implies blurred images. Therefore, the test images include occlusions, illumination changes, blur and other typical nuisances that will be encountered while navigating with a mobile robot. Figure 7 presents sample test images. Finally, this dataset also includes background images, i.e., images that do not contain objects to be recognized, shown in Figure 8.

Several experiments were executed using the typical household objects dataset described. Each experiment consists of a two processing phases: the training of the RL and the execution phase, where the training quality can be verified. To train the RL, we used 40 test images, from which approximately 160 images containing objects were segmented (using the algorithm in Section 3) and previously classified. Sample of the segmented images are presented in Figure 9. Furthermore, 360 background images, also resulting from the segmentation process, were used. The training was performed according to the algorithm described in the previous section.

To evaluate the result of the learning process statistical validation method called Leave-One-Out was used. Using this method, the RL selection module was trained with all the test images but one, and then the image left out (that can contain several ROI with objects and background) is used to test the result of the learning. This

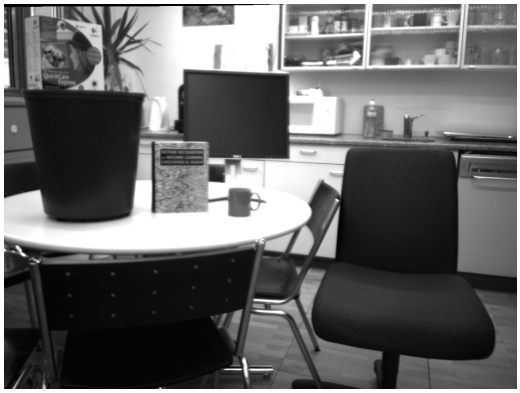

Fig. 7 Test image from the dataset.

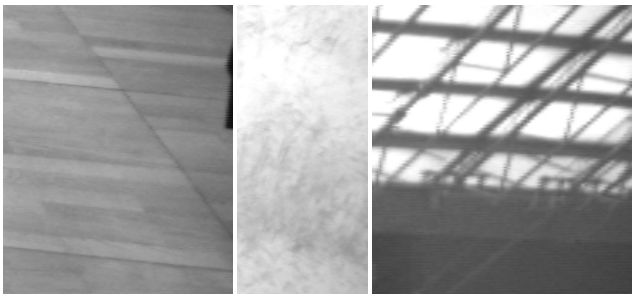

Fig. 8 Background images from the dataset.

test phase corresponds to the execution phase, which can be used on the real robot during on-line exploration of the environment, and its working diagram is presented in Figure 10.

Six different experiments were conducted, using three different combinations of image attributes as space state descriptors and two different image sizes (the image original size and a 10 by 10 pixels reduced size image). The combinations of image attributes used as space state are: mean and standard deviation of the image intensity (MS); mean and standard deviation of the image intensity plus entropy of the image (MSE); and mean and standard deviation of the image intensity plus the number of interest points detected by the Difference of Gaussians operator (MSI).

The rewards used during the learning phase were computed using a set of training images. Figure 11 shows part of the reward table built for the first stage of the first experiment (MS). It is a 100 x 130 figure, where the lines represents the possible values of mean intensity of the image, and columns represents 100 possible values in standard deviation of the image intensity (the MS space). There are three kinds of states, marked in the image by: “X”, which indicates that in the training set exists an image with this combination of mean and std dev values, “·”, which corresponds to points which represents images that does not contain objects (backgrounds) and the

Fig. 9 Objects segmented from test images of the dataset.

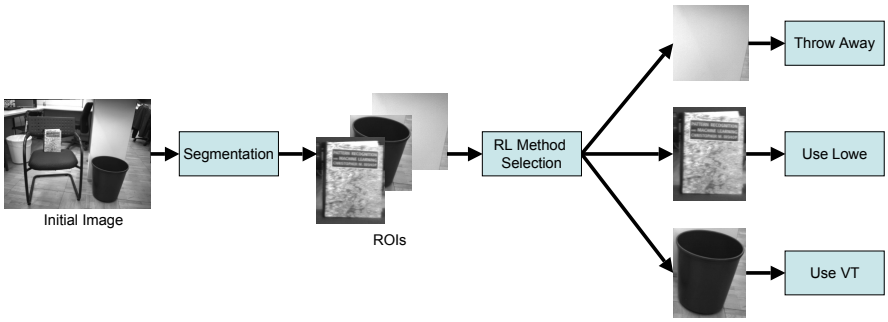

Fig. 10 Execution Phase of the system.

rest of the space, which is left without any marking and correspond to points in the MS state where there is no information about it. As it can be seen, this is a sparse image.

During the learning phase (described in Section 4), if the learning agent reaches a state labeled as "X", it receives the value +10 and if it reaches a state labeled as "·", it receives a reward of -10. Otherwise the reward is zero.

Figure 12 shows the results of applying the RL algorithm during the learning phase. As it can be seen, the classification was spread over to states where there are no prior examples, allowing the classification of other images. This table is the one used during the execution phase.

Tables 1 and 2 present the results obtained for the six experiments. The first line of Table 1 shows the percentage times that the agent correctly choose to discard a background image, and the second line shows the percentage of times the agent correctly choose to use the Lowe algorithm, instead of the Vocabulary Tree one. The columns in this table present the results for the six experiments, the first three

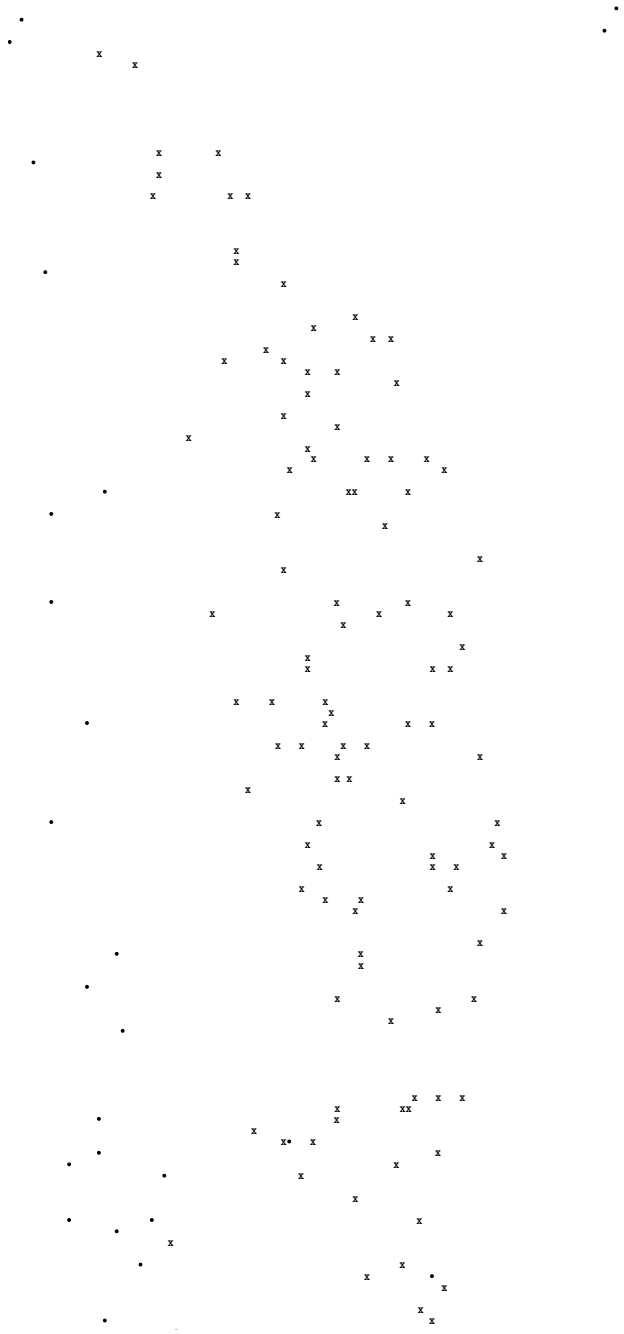

Fig. 11 Image of the reward table built for the first experiment (Mean and Standard deviation of image intensity space).

Fig. 12 Classification table learned in the first experiment.

using the original image and, from the fourth to sixth column, showing the results for the reduced size image. The last column shows the percentage of times a human expert takes the correct action. Table 2 is similar to Table 1, but shows the classification error. The first line shows the percentage of images discarded as background, when they should be analyzed, and line two presents the number of times the Lowe algorithm is chosen, when the correct one is the Vocabulary Tree.

Table 1 Correctly classified images (percentage)

	Full Img			Small Img			Expert
	MS	MSE	MSI	MS	MSE	MSI	
Back	91.9	100.0	98.0	92.6	100.0	98.9	100.0
Lowe	84.5	100.0	44.4	76.0	98.4	38.1	93.2

Table 2 Incorrect classification (percentage)

	Full Img			Small Img			Expert
	MS	MSE	MSI	MS	MSE	MSI	
Back	12.8	1.8	14.2	20.4	2.4	25.3	8.2
Lowe	11.6	1.9	7.9	15.8	1.9	9.9	10.8

The results shows that the use Reinforcement Learning to decide which algorithm should be used to recognize objects yields good results, for all different combinations of image attributes used space state descriptors, performing better than a human expert in some cases.

These tables also show that the best combination of attributes was mean and standard deviation of the image intensity plus entropy of the image (MSE), which presented very good results for original size images as well as reduced size ones. On the other hand, the use of the number of interest points detected by the Difference of Gaussians operator as space state did not produce good results, failing to choose Lowe's algorithm more than half of the time.

Reinforcement Learning algorithms were implemented in C and Computer Vision algorithms were implemented in C++ using the OpenCV library (Bradski, 2000) and Matlab. Experiments were executed on a Pentium 4 Computer running Ubuntu Linux and a PowerMac G4 running Mac OS X. The Reinforcement Learning parameters used in the experiments were: the learning rate $\alpha = 0.1$ and the discount factor $\gamma = 0.9$. Values in the Q table were randomly initiated.

6 Conclusion

Reinforcement Learning has been widely used in the Computer Vision field: to optimize the performance of active vision systems; to decide the focus of attention ; to learn how to move a camera; to optimize parameters of existing and new com-

puter vision algorithms, such as thresholds, contrast and internal parameters; and for constructing object segmentation and recognition strategies by combining CV algorithms. Results have been usually very good, but most of the times limited to a very specific image domain.

In this paper we presented a method that uses Reinforcement Learning to decide which algorithm should be used to recognize objects seen by a mobile robot in an indoor environment, based on simple attributes extracted on-line from the images, such as mean intensity and intensity deviation. Another important contribution of this work is a method that allows the use of a Reinforcement Learning algorithm as a Classifier.

The results obtained shows that the use Reinforcement Learning to decide which algorithm should be used to recognize objects yields good results, performing better than a human expert in some cases. To the best of our knowledge, there is no similar approach using automatic selection of algorithms for object recognition.

Future works includes testing other image attributes that can be used as the system's state, for example, the results of applying Gabor Filters in the images, which may give us clue to the existing texture in the image (Fogel and Sagi, 1989; Weldon et al, 1996). The use of other RL algorithms such as SARSA (Rummery and Niranjan, 1994) and Q (λ)(Peng and Williams, 1996) and the study of the use of this technique in others application domains are also other interesting directions of research.

Acknowledgements This work has been partially funded by the FI grant and the BE grant from the AGAUR, the 2005-SGR-00093 project, supported by the Generalitat de Catalunya, the MID-CBR project grant TIN 2006-15140-C03-01 and FEDER funds. Reinaldo Bianchi is supported by CNPq grant 201591/2007-3.

References

Bhanu B, Peng J (2000) Adaptive integrated image segmentation and object recognition. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 30(4):427–441, DOI 10.1109/5326.897070

Borotschnig H, Paletta L, Prantl M, Pinz A (1999) Appearance-based active object recognition. Image and Vision Computing 18(9):715–728

Bradski G (2000) The OpenCV Library. Dr Dobb's Journal of Software Tools pp 120–125

Canny J (1986) A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell 8(6):679–698

Csurka G, Dance C, Fan L, Willamowski J, Bray C (2004) Visual categorization with bags of keypoints. In: Workshop on Statistical Learning in Computer Vision, ECCV, pp 1–22

Darrell T (1998) Reinforcement learning of active recognition behaviors," interval research technical report 1997-045. (http://www.interval.com/papers/1997-045 -

portions of this paper previously appeared. In: in Advances in Neural Information Processing Systems 8, (NIPS '95, MIT Press, pp 858–864

Darrell T, Pentland A (1996a) Active gesture recognition using learned visual attention. In: Touretzky DS, Mozer MC, Hasselmo ME (eds) Advances in Neural Information Processing Systems, The MIT Press, vol 8, pp 858–864

Darrell T, Pentland A (1996b) Active gesture recognition using partially observable markov decision processes. Pattern Recognition, 1996, Proceedings of the 13th International Conference on 3:984–988 vol.3, DOI 10.1109/ICPR.1996.547315

Draper BA, Bins J, Baek K (1999) ADORE: Adaptive object recognition. In: International Conference on Vision Systems, pp 522–537

Fogel I, Sagi D (1989) Gabor filters as texture discriminator. Biological Cybernetics 61(2):103–113, DOI http://dx.doi.org/10.1007/BF00204594, URL http://dx.doi.org/10.1007/BF00204594

Haralick RM, Sternberg SR, Zhuang X (1987) Image analysis using mathematical morphology. IEEE Trans Pattern Anal Mach Intell 9(4):532–550

Hossain I, Liu J, You J (1999) Tropical cyclone pattern recognition for intensity and forecasting analysis from satellite imagery. Systems, Man, and Cybernetics, 1999 IEEE SMC '99 Conference Proceedings 1999 IEEE International Conference on 6:851–856 vol.6, DOI 10.1109/ICSMC.1999.816663

Lowe D (2004) Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2):91–110

Minut S, Mahadevan S (2001) A reinforcement learning model of selective visual attention. In: AGENTS '01: Proceedings of the fifth international conference on Autonomous agents, ACM, New York, NY, USA, pp 457–464, DOI http://doi.acm.org/10.1145/375735.376414

Mitchell T (1997) Machine Learning. McGraw Hill, New York

Nistér D, Stewénius H (2006) Scalable recognition with a vocabulary tree. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, vol 2, pp 2161–2168

Paletta L, Pinz A (2000) Active object recognition by view integration and reinforcement learning. Robotics and Autonomous Systems 31(1–2):71–86

Paletta L, Prantl M, Pinz A (1998) Reinforcement learning of 3-d object recognition from appearance. In: CONALD'98: Proceedings of the 1998 Conference on Automated Learning and Discovery

Paletta L, Fritz G, Seifert C (2005) Q-learning of sequential attention for visual object recognition from informative local descriptors. In: ICML '05: Proceedings of the 22nd International Conference on Machine Learning, ACM, New York, NY, USA, pp 649–656, DOI http://doi.acm.org/10.1145/1102351.1102433

Paris S, Kornprobst P, Tumblin J, Durand F (2007) A gentle introduction to bilateral filtering and its applications. In: SIGGRAPH '07: ACM SIGGRAPH 2007 courses, ACM, New York, NY, USA, p 1, DOI http://doi.acm.org/10.1145/1281500.1281602

Peng J, Bhanu B (1998a) Closed-loop object recognition using reinforcement learning. Pattern Analysis and Machine Intelligence, IEEE Transactions on 20(2):139–154, DOI 10.1109/34.659932

Peng J, Bhanu B (1998b) Delayed reinforcement learning for adaptive image segmentation and feature extraction. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 28(3):482–488, DOI 10.1109/5326.704593

Peng J, Williams RJ (1996) Incremental multi-step q-learning. Mach Learn 22(1-3):283–290, DOI http://dx.doi.org/10.1007/BF00114731

Ramisa A, Vasudevan S, Scaramuzza D, de Mantaras RL, Siegwart R (2008) A tale of two object recognition methods for mobile robots. In: Gasteratos A, Vincze M, Tsotsos JK (eds) International Conference on Computer Vision Systems, Springer, Lecture Notes in Computer Science, vol 5008, pp 353–362

Rummery GA, Niranjan M (1994) On-line Q-learning using connectionist systems. Tech. Rep. CUED/F-INFENG/TR 166, Cambridge University Engineering Department

Sahba F, Tizhoosh HR, Salama MMMA (2008) A reinforcement agent for object segmentation in ultrasound images. Expert Syst Appl 35(3):772–780, DOI http://dx.doi.org/10.1016/j.eswa.2007.07.057

Shokri M, Tizhoosh HR (2003) Using reinforcement learning for image thresholding. Electrical and Computer Engineering, 2003 IEEE CCECE 2003 Canadian Conference on 2:1231–1234 vol.2, DOI 10.1109/CCECE.2003.1226121

Shokri M, Tizhoosh HR (2004) Q(lambda)-based image thresholding. In: CRV '04: Proceedings of the 1st Canadian Conference on Computer and Robot Vision, IEEE Computer Society, Los Alamitos, CA, USA, pp 504–508, DOI http://doi.ieeecomputersociety.org/10.1109/CCCRV.2004.1301490

Shokri M, Tizhoosh HR (2008) A reinforcement agent for threshold fusion. Appl Soft Comput 8(1):174–181, DOI http://dx.doi.org/10.1016/j.asoc.2006.12.003

Sivic J, Schaffalitzky F, Zisserman A (2006) Object Level Grouping for Video Shots. International Journal of Computer Vision 67(2):189–210

Sutton R, Barto AG (1998) Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA

Taylor GW (2004) Reinforcement Learning for Parameter Control of Image-Based Applications. MSc Thesis, University of Waterloo, Ontario, Canada

Tizhoosh H, Taylor G (2006) Reinforced contrast adaptation. International Journal of Image and Graphics 6(3):377–392

Tomasi C, Manduchi R (1998) Bilateral filtering for gray and color images. In: ICCV '98: Proceedings of the Sixth International Conference on Computer Vision, IEEE Computer Society, Washington, DC, USA, p 839

Watkins CJCH (1989) Learning from Delayed Rewards. PhD Thesis, University of Cambridge

Weldon TP, Higgins WE, Dunn DF (1996) Efficient gabor filter design for texture segmentation. Pattern Recognition 29(12):2005–2015, DOI http://dx.doi.org/10.1016/S0031-3203(96)00047-7, URL http://dx.doi.org/10.1016/S0031-3203(96)00047-7

Whitehead SD, Ballard DH (1991) Learning to perceive and act by trial and error. Mach Learn 7(1):45–83, DOI http://dx.doi.org/10.1023/A:1022619109594

Wolf C, Jolion JM (2003) Extraction and recognition of artificial text in multimedia documents. Pattern Anal Appl 6(4):309–326, DOI http://dx.doi.org/10.1007/s10044-003-0197-7

Yin PY (2002) Maximum entropy-based optimal threshold selection using deterministic reinforcement learning with controlled randomization. Signal Process 82(7):993–1006, DOI http://dx.doi.org/10.1016/S0165-1684(02)00203-7