Image and Vision Computing 28 (2010) 1080–1097

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

The more you learn, the less you store: Memory-controlled incremental SVM for visual place recognition

Andrzej Pronobis [a,*], Luo Jie [b,c], Barbara Caputo [b]

[a] CAS/CVAP, The Royal Institute of Technology (KTH), SE-100 44 Stockholm, Sweden
[b] Idiap Research Institute, Rue Marconi 19, CH-1920 Martigny, Switzerland
[c] Swiss Federal Institute of Technology in Lausanne (EPFL), CH-1015 Lausanne, Switzerland

ARTICLE INFO

Article history:
Received 19 September 2008
Received in revised form 27 January 2010
Accepted 29 January 2010

Keywords:
Incremental learning
Knowledge transfer
Support vector machines
Place recognition
Visual robot localization

ABSTRACT

The capability to learn from experience is a key property for autonomous cognitive systems working in realistic settings. To this end, this paper presents an SVM-based algorithm, capable of learning model representations incrementally while keeping under control memory requirements. We combine an incremental extension of SVMs [43] with a method reducing the number of support vectors needed to build the decision function without any loss in performance [15] introducing a parameter which permits a user-set trade-off between performance and memory. The resulting algorithm is able to achieve the same recognition results as the original incremental method while reducing the memory growth. Our method is especially suited to work for autonomous systems in realistic settings. We present experiments on two common scenarios in this domain: adaptation in presence of dynamic changes and transfer of knowledge between two different autonomous agents, focusing in both cases on the problem of visual place recognition applied to mobile robot topological localization. Experiments in both scenarios clearly show the power of our approach.

1. Introduction

Many recent advances in fields such as computer vision and robotics have been driven by the ultimate goal of creating artificial cognitive systems able to perform human-like tasks. Several attempts have been made to create integrated cognitive architectures and implement them, for instance, on mobile robots [2,23,1,3]. The ability to learn and interpret complex sensory information based on the previous experience, inherently connected with cognition, has been recognized as crucial and vastly researched [42,40,33]. In most cases, the recognition systems used are trained offline, i.e. they are based on batch learning algorithms. However, in the real, dynamic world, learning cannot be a single act. It is simply not possible to create a static model which could explain all the variability observed over time. Continuous information acquisition and exchange, coupled with an ongoing learning process, is necessary to provide a cognitive system with a valid world representation.

In artificial autonomous agents constrained by limited resources (such as mobile robots), continuous learning must be performed in an incremental fashion. It is obviously not feasible to rebuild the internal model from scratch every time new information arrives, neither it is possible to store all the previously acquired data for that purpose. The model must be updated and the updating process must have certain properties. First, the knowledge representation must remain compact and free from redundancy to fit into the limited memory and maintain a fixed computational complexity. We call this property *controlled memory growth*. Second, in the continuous learning scenario, a model cannot grow forever even though new information is constantly arriving. Thus, the updating process should be able to gradually filter out unnecessary information. We call this property *forgetting capability*.

Discriminative methods have become widely popular for visual recognition, achieving impressive results on several applications [47,20,14]. Within discriminative classifiers, SVM techniques provide powerful tools for learning models with good generalization capabilities; in some domains like object and material categorization, SVM-based algorithms are state of the art [7,17]. This makes it worth it to investigate whether it is possible to perform continuous learning with this type of methods. Several incremental extensions of SVMs have been proposed in the machine learning community [13,8,43,35]. Between these methods, the approximate techniques [13,43] seem better suited for visual recognition because, at each incremental step, they discard non-informative training vectors, thus reducing the memory requirements. Other methods, such as [8,35], instead require to store in memory all the training data, eventually leading to a memory explosion; this makes them unfit for real-time autonomous systems.

* Corresponding author. Tel.: +46 8 790 6725; fax: +46 8 790 6725.
E-mail addresses: pronobis@csc.kth.se (A. Pronobis), jluo@idiap.ch (L. Jie), bcaputo@idiap.ch (B. Caputo).

0262-8856/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2010.01.015

This paper presents an SVM-based incremental method which performs like the batch algorithm while reducing the memory requirements. We combine an approximate technique for incremental SVM [43] with an exact method that reduces the number of support vectors needed to build the decision function without any loss in performance [15]. This results in an algorithm performing as the original incremental method with a reduction in the memory requirements. We then present an extension of the method for the exact simplification of the support vector solution [15]. We introduce a parameter that links the performance of an SVM to the amount of vectors that is possible to discard. This allows a user-set trade-off between performance and memory reduction.

We evaluate the suitability of our method for autonomous cognitive systems in two challenging scenarios: adaptation in presence of dynamic changes and transfer of knowledge between autonomous agents. In both cases, we concentrate on the problem of visual place recognition applied to mobile robot topological localization. The problem is important from the point of view of engineering cognitive systems, as it allows to tie semantics with space representations and provides solutions for typical problems with purely metric localization. However, it is also a challenging recognition problem as it requires processing of large amounts of high-dimensional visual information which is noisy and dynamic in nature. In this context, the memory and computational efficiency become one of the most important properties of the learning algorithm determining the design choice.

In our considerations, we first focus on the scenario in which the incremental learning is used to provide adaptability to different types of variations observed in real-world environments. In our previous work [39,37], we presented a purely appearance-based model able to cope with illumination and pose changes, and we showed experimentally that it could achieve satisfactory performances when considering short time intervals between the acquisition of the training and testing data. Nevertheless, a room's appearance is doomed to change dramatically over time because it is used: chairs are pushed around, objects are taken in/out of drawers, furniture and paintings are added, or changed, or re-arranged; and so forth. As it is not possible to predict a priori how a room is going to change, the only possible strategy is to update the representation over time, learning incrementally from the new data recorded during use.

As a second scenario, we consider the case when a robot, proficient in solving the place recognition task within a known environment, transfers its visual knowledge to another robotic platform with different characteristics, which is a tabula rasa. The ability to transfer knowledge between different domains enables humans to learn efficiently from small number of examples. This observation inspired robotics and machine learning researchers to search for algorithms able to exploit prior knowledge so to improve performance of artificial learners and speed up the learning process. To tackle this problem, it is necessary an efficient way of exploiting the knowledge transferred from a different platform as well as updating the internal representation when new training data are available. The knowledge transfer scheme should be adaptive and privilege newest data so to prevent from accumulating outdated information. Finally, the solution obtained starting from a transferred model should gradually converge to the one learned from scratch, not only in terms of performance on a task but also of required resources (e.g. memory).

To achieve these goals, we used our memory-controlled incremental SVM and we evaluated its performance in terms of accuracy, memory growth, complexity and forgetting capability. We compare the results obtained by our method with those achieved by the batch algorithm and by two other incremental extensions of SVMs, one approximate (the fixed-partition incremental SVM [43]) and one exact (online independent SVM, [35]). We evaluated the algorithms on a visual place recognition database acquired using two mobile robot platforms [39], which we extended with new data acquired 6 months later using the same hardware. Then, we confirmed the results on another database acquired in a different environment and using different hardware [38]. To test the adaptability of the recognition system, we performed topological localization experiments under realistic long-term variations. To test the knowledge transfer capabilities, we performed experiments in case of which visual knowledge captured in the SVM model was gradually exchanged between the two mobile robot platforms. The experiments clearly show the power of our approach in both scenarios, while illustrating the need for incremental solutions in artificial cognitive systems.

The rest of the paper is organized as follows: after a review of related work (Section 2), Section 3 gives our working definition of visual place recognition for robot localization. Section 4 reviews SVMs, it introduces the memory-controlled incremental SVM algorithm, which will constitute a building block of the adaptive place recognition system and a base for our knowledge transfer technique, and it briefly describes two other incremental extensions of SVMs against which we will benchmark our approach. Section 5 describes our experimental setup; Section 7 concentrates on the adaptation problem and presents experimental evaluation of the algorithms in this context. Finally, Section 8 gives details of our approach to the transfer of knowledge and shows its effectiveness with a set of experiments. The paper concludes with a summary and possible directions for future work.

2. Related work

In the last years, the need for solutions to such problems as robustness to long-term dynamic variations or transfer of knowledge is more and more acknowledged. In [40], the authors tried to deal with long-term visual variations in indoor environments by combining information acquired using two sensors of different characteristics. In [49], the problem of invariance to seasonal changes in appearance of an outdoor environment is addressed. Clearly, adaptability is a desirable property of a recognition system. At the same time, Thrun and Mitchell [46,32] studied the issue of exchanging knowledge related to different tasks in the context of artificial neural networks and argued for the importance of knowledge transfer schemes for lifelong robot learning. Several attempts to solve the problem have also been made from the perspective of reinforcement learning, including the case of transferring learned skills between different RL agents [29,21].

The work conducted in the fields of cognitive robotics and vision stimulated the research in the machine learning community directed towards developing extensions for algorithms that were commonly used due to their superior performance but were missing the ability to be trained incrementally. As a result, methods such as incremental PCA have been invented and successfully applied e.g. for mobile robot localization [4,11]. As it was already mentioned, several incremental extensions have been introduced also for support vector machines [13,8,43]. Between these methods, the approximate techniques [13,43] seem better suited for visual recognition because, at each incremental step, they discard non-informative training vectors, thus reducing the memory requirements. Other methods, such as [8,35], or simple KNN-based solutions, instead require to store in memory all the training data, eventually leading to a memory explosion. This limits their usefulness for complex real-world problems involving continuous learning of visual patterns.

Despite the fact that the approximate incremental SVM extensions allow to reduce the amount of data stored during the learning process, there is no guarantee that the continuously updated mod-

el will not grow forever. Additionally, the results of experiments that can be found in the literature do not give a clear answer if it is possible to apply such methods for complex problems such as visual place recognition or transfer of visual knowledge.

3. Visual place recognition for robot localization

In this section, we give our working definition of visual place recognition, explaining how it can be applied to mobile robot topological localization. We define a place as a nameable segment of a real-world environment, uniquely identifiable because of its specific functionality and/or appearance. Examples of places, according to this definition, are a kitchen, an office, a corridor, and so forth. We adopt the appearance-based paradigm, and we assume that a realistic scene can be represented by a visual descriptor without any loss of discriminative information. We consider a fully supervised, incremental learning scenario: we assume that, at each incremental step, every room is represented by a collection of images which capture its visual appearance under different viewpoints, at a fixed time and illumination setting. During testing, the algorithm is presented with images of the same rooms, acquired under similar viewpoints but possibly under different illumination conditions and after some time, with a time range going from some minutes to several months. The goal is to recognize correctly each single image seen by the system. Fig. 1 illustrates the approach.

A typical application for an indoor place recognition system is topological robot localization. The localization problem is vastly researched. This resulted, over the years, in a broad range of approaches spanning from purely metric [19,12,52,16], to topological [48,30,40], and hybrid [45,6]. Traditionally, sonar and/or laser have been the sensory modalities of choice [34,30]. Yet, the inability to capture many aspects of complex realistic environments leads to the problem of perceptual aliasing [24], and greatly limits the usefulness of such methods for semantic mapping. Recent advances in vision have made this modality emerge as a natural and viable solution for localization problems. Vision provides richer sensory input allowing for better discrimination. It opens new possibilities for building cognitive systems, actively relying on semantic context. Not unimportant is the cost effectiveness, portability and popularity of visual sensors. As a result, despite the complexity of the problem, this research line is attracting more and more attention, and several methods have been proposed using vision alone [41,48,47,37,42], or combined with more traditional range sensors [22,44,40].

Our visual place recognition system uses SVM-based discriminative place models trained on global and local image features. These features are described in details in Section 5. The classification algorithm is introduced in Section 4. In our experiments, we always used only a single image as input for the recognition system. This makes the recognition problem harder, but also it makes it possible to perform global localization where no prior knowledge about the position is available (e.g. in case of the kidnapped robot problem). Spatial or temporal filtering can be used together with the presented method to enhance performance.

4. Memory-controlled incremental SVM

This section describes our algorithmic approach to incremental learning of visual place models. We propose a fully supervised, SVM-based method with controlled memory growth that tends to privilege newest information over older data. This leads to a system able to adapt over time to the natural changes of a real-world setting, while maintaining a limited memory size and computational complexity.

The rest of this section describes the basic principles of support vector machines (Section 4.1), a popular incremental extension of the basic algorithm (Section 4.2), our memory-controlled version of incremental SVM (Section 4.3) and an exact method based on a similar intuition (Section 4.4), with which we will compare our approach.

4.1. SVM: the batch algorithm

Consider the problem of separating the set of training data $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)$ into two classes, where $\boldsymbol{x}_i \in \Re^N$ is a feature vector and $y_i \in \{-1, +1\}$ its class label (for multi-class extensions, we refer the reader to [10,50]). If we assume that the two classes can be linearly separated when mapped to some higher dimensional Hilbert space $\mathscr{H}$ by $\boldsymbol{x} \to \Phi(\boldsymbol{x}) \in \mathscr{H}$ (see [10,50] for solutions to non-separable cases), the optimal hyperplane is the one which has maximum distance to the closest points in the training set, resulting in a classification function:

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{m} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b\right), \tag{1}$$

where $K(\boldsymbol{x}, \boldsymbol{y}) = \Phi(\boldsymbol{x}) \cdot \Phi(\boldsymbol{y})$ is the kernel function. Most of the α_i's take the value of zero; $\boldsymbol{x}_i$ with non-zero α_i are the support vectors (SV). Different kernel functions correspond to different similarity measures. Choosing a suitable kernel can therefore have a strong impact on the performance of the classifier. Based on results reported in the literature [39], here we used the two following kernels:

- The χ^2 kernel [5] for histogram-like global descriptors:

$$K(\boldsymbol{x}, \boldsymbol{y}) = \exp\{-\gamma \chi^2(\boldsymbol{x}, \boldsymbol{y})\}, \quad \chi^2(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{N} \frac{(x_i - y_i)^2}{x_i + y_i},$$

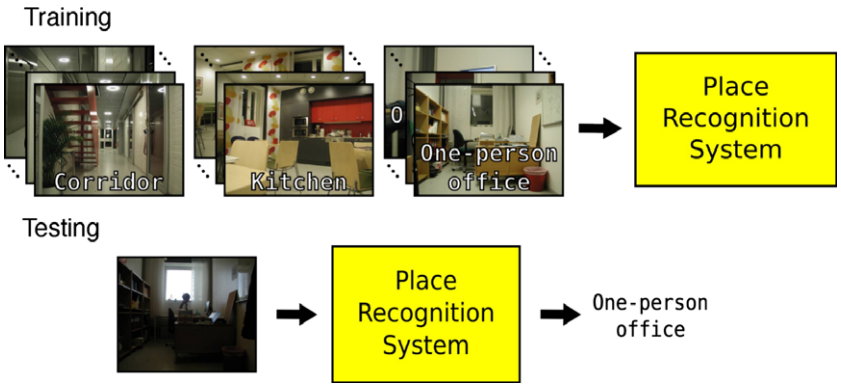


Fig. 1. A schematic representation of our visual place recognition system for robot localization.

- The matching kernel [51] for local features:

$$K(\boldsymbol{L}_h, \boldsymbol{L}_k) = \frac{1}{n_h} \sum_{j_h=1}^{n_h} \max_{j_k=1,\ldots,n_k} \left\{ K_l\left(\boldsymbol{L}_h^{j_h}, \boldsymbol{L}_k^{j_k}\right) \right\},$$

where $\boldsymbol{L}_h$, $\boldsymbol{L}_k$ are local feature sets and $\boldsymbol{L}_h^{j_h}$, $\boldsymbol{L}_k^{j_k}$ are two single local features. The sum is always calculated over the smaller set of local features and only some fixed amount of best matches is considered in order to exclude outliers. The local feature similarity kernel K_l can be any Mercer kernel. We used the RBF kernel based on the Euclidean distance for the SIFT [27] features:

$$K_l\left(\boldsymbol{L}_h^{j_h}, \boldsymbol{L}_k^{j_k}\right) = \exp\left\{-\gamma \|\boldsymbol{L}_h^{j_h} - \boldsymbol{L}_k^{j_k}\|^2\right\}.$$

4.2. SVM: an incremental extension

Among the incremental SVM extensions proposed so far [43,13,8], approximate methods seem to be the most suitable for visual recognition, because they discard a significant amount of the training data at each incremental step. Exact methods instead need to retain all training samples in order to preserve the convexity of the solution at each incremental step. As a consequence, they require huge amounts of memory when employed in realistic, continuous learning scenario as the one we consider here. Approximate methods avoid this problem by sacrificing the guaranteed optimality of the solution. Still, several studies showed that they generally achieve performances very similar to those obtained by an SVM trained on the complete data set (see [13] and references therein), because at each incremental step the algorithm remembers the essential class boundary information regarding the data seen so far (in form of support vectors). This information contributes properly to generate the classifier at the next iteration.

Once a new batch of data is loaded into memory, there are different possibilities for performing the update of the current model, which might discard a part of the new data according to some fixed criteria [13,43]. For all the techniques, at each step only the learned model from the data previously seen (preserved in form of SV) is kept in memory. In this paper we will consider the fixed-partition method [43]. Here the training data set is partitioned in batches of some size k:

$$\boldsymbol{T} = \{\boldsymbol{T}_1, \boldsymbol{T}_2, \ldots, \boldsymbol{T}_n\},$$

with $\boldsymbol{T}_i = \left\{\left(\boldsymbol{x}_j^i, y_j^i\right)\right\}_{j=1}^{k}$. At the first step, the model is trained on the first batch of data $\boldsymbol{T}_1$, obtaining a classification function

$$f_1(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m_1} \alpha_i^1 y_i^1 K(\boldsymbol{x}_i^1, \boldsymbol{x}) + b^1\right). \tag{2}$$

At the second step, a new batch of data is loaded into memory and added to the current set of support vectors; then, the *new* training set becomes

$$\boldsymbol{T}_2^{inc} = \{\boldsymbol{T}_2 \cup \boldsymbol{SV}_1\}, \quad \boldsymbol{SV}_1 = \left\{\left(\boldsymbol{x}_i^1, y_i^1\right)\right\}_{i=1}^{m_1},$$

where $\boldsymbol{SV}_1$ are the support vectors learned at the first step. The new classification function will be:

$$f_2(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{m_2} \alpha_i^2 y_i^2 K(\boldsymbol{x}_i^2, \boldsymbol{x}) + b^2\right).$$

Thus, as new batches of data points are loaded into memory, the existing support vector model is updated, so to generate the classifier at that incremental step. The method is illustrated in Fig. 2. Note that this incremental method can be seen as an approximation of the chunking technique used for training SVM [10,50]. Indeed, the chunking algorithm is an exact decomposition which iterates through the training set to select the support vectors. The fixed-partition incremental method instead scan through the training data just once, and once discarded, does not consider them anymore. The fixed-partition incremental algorithm has been tested on several benchmark databases commonly used in the machine learning community [13], obtaining good performances comparable to the batch algorithm and other approximate methods. An open issue is that in principle there is no limitation to the memory growth. Indeed, several experimental evaluations show that, while approximate methods generally achieve classification performances equivalent to those of batch SVM, the number of SV tends to grow proportionally to the number of incremental steps (see [13] and references therein).

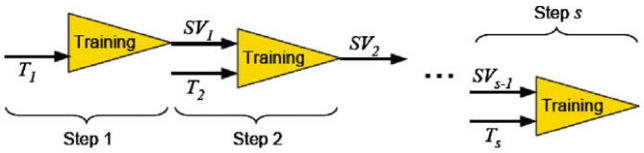


Fig. 2. The fixed-partition incremental SVM algorithm.

4.3. Memory-controlled incremental SVM

The core idea of the memory-controlled incremental SVM is that the set of support vectors $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^m$ in Eq. (1) is not guaranteed to be linearly independent. Based on this observation, it is possible to reduce the number of support vectors of a trained classifier, eliminating those which can be expressed as a linear combination of the others in the feature space, as proposed in [15] for reducing the complexity of the SVM solution. By updating the weights accordingly, it is ensured that the decision function is exactly the same as the original one. More specifically, let us suppose that the first r support vectors are linearly independent, and the remaining $m - r$ depend linearly on those in the feature space: $\forall j = r+1, \ldots, m,\ \boldsymbol{x}_j \in span\{\boldsymbol{x}_i\}_{i=1}^r$. Then it holds

$$K(\boldsymbol{x}, \boldsymbol{x}_j) = \sum_{i=1}^{r} c_{ij} K(\boldsymbol{x}, \boldsymbol{x}_i), \tag{3}$$

and the classification function (1) can be rewritten as

$$f(\boldsymbol{x}) = \operatorname{sgn}\left(\sum_{i=1}^{r} \alpha_i y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + \sum_{j=r+1}^{m} \alpha_j y_j \sum_{i=1}^{r} c_{ij} K(\boldsymbol{x}, \boldsymbol{x}_i) + b\right). \tag{4}$$

If we define the coefficients γ_{ij} such that $\alpha_j y_j c_{ij} = \alpha_i y_i \gamma_{ij}$ and $\gamma_i = \sum_{j=r+1}^{m} \gamma_{ij}$, then Eq. (4) can be written as

$$\begin{aligned} f(\boldsymbol{x}) &= \operatorname{sgn}\left(\sum_{i=1}^{r} \alpha_i y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + \sum_{i=1}^{r} \alpha_i y_i \sum_{j=r+1}^{m} \gamma_{ij} K(\boldsymbol{x}, \boldsymbol{x}_i) + b\right) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{r} \alpha_i (1+\gamma_i) y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b\right) \\ &= \operatorname{sgn}\left(\sum_{i=1}^{r} \widehat{\alpha}_i y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b\right), \end{aligned} \tag{5}$$

where

$$\widehat{\alpha}_i = \alpha_i (1+\gamma_i) = \alpha_i \left(1 + \sum_{j=r+1}^{m} \frac{\alpha_j y_j c_{ij}}{\alpha_i y_i}\right).$$

The α_i coefficients can be pre-multiplied by the class labels $\alpha_i' = \alpha_i y_i$ which results in a simple equation that can be used to obtain the weights of the reduced classifier:

$$\widehat{\alpha}'_i = \begin{cases} \alpha'_i + \sum_{j=r+1}^{m} \alpha'_j c_{ij} & \text{for } i = 1, 2, \ldots, r, \\ 0 & \text{for } i = r+1, r+2, \ldots, m. \end{cases} \quad (6)$$

Thus, the resulting classification function (Eq. (5)) requires now $m - r$ less kernel evaluations than the original one.

The linearly independent subset of the support vectors as well as the coefficients c_{ij} can be found by applying methods from linear algebra to the support vector matrix given by

$$\boldsymbol{K} = \begin{bmatrix} K(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & K(\boldsymbol{x}_1, \boldsymbol{x}_m) \\ \vdots & \ddots & \vdots \\ K(\boldsymbol{x}_m, \boldsymbol{x}_1) & \cdots & K(\boldsymbol{x}_m, \boldsymbol{x}_m) \end{bmatrix}, \quad (7)$$

We employ the QR factorization with column pivoting [18] for this purpose. The QR factorization with column pivoting algorithm is a widely used method for selecting the independent columns of a matrix. The algorithm allows to reveal the numerical rank of the matrix with respect to a parameter τ, which acts as a threshold in defining the condition of linear dependence. Additionally, it performs a permutation of the columns of the matrix so that they are ordered according to the degree of their relative linear independence. Consequently, if for a given value of τ the rank of the matrix is r, then the linearly independent columns will occupy the first r positions.

The QR factorization with column pivoting of the matrix $\boldsymbol{K} \in \Re^{m \times m}$ is given by

$$\boldsymbol{K\Pi} = \boldsymbol{QR}, \quad (8)$$

where $\boldsymbol{\Pi} \in \Re^{m \times m}$ is a permutation matrix, $\boldsymbol{Q} \in \Re^{m \times m}$ is orthogonal, and $\boldsymbol{R} \in \Re^{m \times m}$ is upper triangular. If we assume that the rank of the matrix K with respect to the parameter τ equals r, then the matrices can be decomposed as follows:

$$[\boldsymbol{K}_1 \quad \boldsymbol{K}_2] = [\boldsymbol{Q}_1 \quad \boldsymbol{Q}_2] \begin{bmatrix} \boldsymbol{R}_{11} & \boldsymbol{R}_{12} \\ \boldsymbol{0} & \boldsymbol{R}_{22} \end{bmatrix}, \quad (9)$$

where the columns of $\boldsymbol{K}_1 \in \Re^{m \times r}$ create a linearly independent set, the columns of $\boldsymbol{K}_2 \in \Re^{m \times m-r}$ may be expressed as a linear combination of the columns of $\boldsymbol{K}_1$, $\boldsymbol{Q}_1 \in \Re^{m \times r}$, $\boldsymbol{Q}_2 \in \Re^{m \times m-r}$, $\boldsymbol{R}_{11} \in \Re^{r \times r}$, $\boldsymbol{R}_{12} \in \Re^{r \times m-r}$, $\boldsymbol{R}_{22} \in \Re^{m-r \times m-r}$.

The products of the QR factorization can be used to obtain the coefficients c_{ij} as follows:

$$\boldsymbol{C} = \begin{bmatrix} c_{1,r+1} & \cdots & c_{1,m} \\ \vdots & \ddots & \vdots \\ c_{r,r+1} & \cdots & c_{r,m} \end{bmatrix} = \boldsymbol{R}_{11}^{-1} \boldsymbol{Q}_1^T \boldsymbol{K}_2. \quad (10)$$

The coefficients together with the permutation matrix $\boldsymbol{\Pi} \in \Re^{m \times m}$ and the number of the linearly independent support vectors r are sufficient to obtain the reduced solution. Using matrix notation, Eq. (6) can be expressed as follows:

$$\begin{cases} \hat{\boldsymbol{\alpha}}'_1 = \boldsymbol{\alpha}'_1 + \boldsymbol{R}_{11}^{-1} \boldsymbol{Q}_1^T \boldsymbol{K}_2 \boldsymbol{\alpha}'_2, \\ \hat{\boldsymbol{\alpha}}'_2 = \boldsymbol{0}. \end{cases} \quad (11)$$

The rank r of the matrix $\boldsymbol{K}$ can be estimated by thresholding $\|\boldsymbol{R}_{22}\|_2$ with the value of the parameter τ. This means that, in practice, the choice of the τ value determines the number of linearly independent support vectors retained by the algorithm. For instance, by choosing a value of τ of 0.1 one will select a number of linearly independent support vectors smaller than by choosing a τ value of 0.01. This has two concrete effects on the algorithm:

(1) As the value of τ increases, the number of support vectors decreases. This means that, by tuning τ, it is possible to reduce the memory requirements and to increase speed during classification.
(2) At the same time, as τ increases, Eq. (5) will become more and more an approximation of the exact solution, because we are considering as linearly dependent vectors that are not. Therefore, we are not able to preserve fully their informative content. Still, we do not lose all the information carried by the discarded support vector $\mathbf{x}_j$, as its weight α_j is used to compute the updated value of the weights $\widehat{\alpha}_i$ for the remaining support vectors. This should result in a graceful decrease of classification performance compared to the optimal solution.

We propose to combine this model simplification with the fixed-partition incremental algorithm, adding the reduction process at each incremental step. We call the new algorithm memory-controlled incremental SVM. It can be illustrated as follows:

(1) **Train**. The algorithm receives the first batch of data $\boldsymbol{T}_1$. It trains an SVM and obtains a set of support vectors $\boldsymbol{SV}_1$.
(2) **Find linearly dependent SVs**. The algorithm finds permutation of $\boldsymbol{SV}_1$ that orders the SVs according to the degree of their linear independence.
(3) **Find** τ. The algorithm searches for the value of τ, $\tau^\star$, that satisfies certain requirements regarding the number of support vectors or estimated performance of the classifier.
(4) **Reduce**. The algorithm computes the reduced solution determined by the chosen $\tau^\star$. After this step, the reduced model contains a subset of the original SVs, $\widehat{\boldsymbol{SV}}_1 = red(\boldsymbol{SV}_1)$, and can be used to classify test data.
(5) **Retrain**. As the new batch of data $\boldsymbol{T}_2$ arrives, step (1) is repeated using as training vectors $\widehat{\boldsymbol{T}}_2^{inc} = \{\boldsymbol{T}_2 \cup \widehat{\boldsymbol{SV}}_1\}$.

For applications that require speed and/or have limited memory requirements, at step (3) of the algorithm, one can tune τ so to obtain at each incremental step a pre-defined maximum number of stored SV. For applications where accuracy is more relevant, one can estimate at each incremental step the τ corresponding to a pre-defined maximum decrease in performance. This can be done on the batch of data $\boldsymbol{T}_i$ at each step, dividing $\boldsymbol{T}_i$ in two subsets and training on one and testing on the other or by applying the leave-one-out strategy. We denote with the symbol Θ the percentage of the original classification rate that is guaranteed to be preserved after the reduction in this case.

In order to apply the method to multi-class problems, we used the one-vs.-one multi-class extension. In a set of preliminary experiments comparing the one-vs.-one and one-vs.-all algorithms, we did not observe significant differences in the behavior of both methods (for further details, we refer the reader to [36]). The one-vs.-one algorithm, given M classes, trains $M(M-1)/2$ two class SVMs, one for each pair of classes. In case of the place recognition experiments, this method obtained smaller training times due to large number of training samples and relatively small number of classes.

4.4. Online independent incremental SVM

The idea to exploit the linear independence in the feature space has also been implemented in an online extension of SVMs, called Online Independent Support Vector Machine (OISVM [35]). OISVM selects incrementally basis vectors that are used to build the solution of the SVM training problem, based upon linear independence in the feature space. Vectors that are linearly dependent on already stored ones are rejected. An incremental minimization algorithm is

employed to find the new minimum of the cost function. This approach reduces considerably the complexity of the solution and therefore the testing time. As OISVM is an exact method, it requires to store all data acquired by the system during its whole life span for the update of the cost function. In many cases (e.g. in case of place recognition), the data samples are multi-dimensional and require a substantial amount of storage. Additionally, the learning algorithm needs to build a gram matrix the size of which is quadratic in the number of training samples. This leads inevitably to a memory explosion when the number of incremental steps grows, as we will show experimentally. Through its heuristics, the memory-controlled algorithm allows to decrease the number of training data samples at each incremental step and thus reduce the memory consumption.

5. Experimental setup

This section describes our experimental setup. We first describe the IDOL2 and COLD-Freiburg databases, on which we will run all the experiments reported in this paper (Sections 5.1 and 5.2), then we briefly describe the feature representations used in the experiments (Section 5.3). Finally, we discuss the performance evaluation measure and parameter selection method (Section 5.4).

5.1. The IDOL2 database

The Image Database for rObot Localization 2 (IDOL2 [28]) database contains 24 image sequences acquired by a perspective camera, mounted on two mobile robot platforms. Both mobile robot platforms, the PeopleBot Minnie and the PowerBot Dumbo, are equipped with cameras. On Minnie the camera is located 98 cm above the floor, whereas on Dumbo its height is 36 cm. Fig. 3 shows both robots and some sample images from the database acquired by the robots from very close viewpoints, illustrating the difference in visual content. These images were acquired under the same illumination conditions and within short time spans.

The robots were manually driven through an indoor laboratory environment and the images were acquired at a rate of 5 fps. Each image sequence consists of 800–1100 frames automatically labeled with one of five different classes (Printer Area [PA], CoRridor [CR], KiTchen [KT], Two-persons Office [TO], and One-person Office [OO]). The labeling is based on the camera's position given by the laser-based localization system proposed in [16]. The acquisition procedure was repeated several times to capture the changes in illumination and varying weather conditions (sunny, cloudy, and night). Also, special care was taken to capture people's activities, change of location for objects and for furniture; for part of the environment (two-persons office) we were able to record a significant change in decoration which occurred over a time span of 6 months. Fig. 4 shows some sample images from the database, illustrating these variations. It is important to note that each single sequence captures the appearance of the considered experimental environment under stable illumination settings and during the short span of time that is required to drive the robot manually around the environment.

The 24 image sequences are divided as follows: for each robot platform and for each type of illumination conditions (cloudy, sunny, night), there are four sequences recorded. Of these four sequences, the first two were acquired six months before the last two. This means that, for every robot we always have subsets of sequences acquired under similar conditions and close in time, as well as subsets acquired under different conditions and distant in time. This makes the database useful for several types of experiments. It is important to note that, even for the sequences acquired within a short time span, variations still exist from everyday activities and viewpoint differences during acquisition. For further details, we refer the reader to [28].

5.2. The COLD-Freiburg database

The COLD-Freiburg database is a collection of image sequences acquired at the Autonomous Intelligent System Laboratory at the University of Freiburg and constitutes a part of the COsy Localization Database (COLD, [38]). The acquisition procedure of the COLD-Freiburg database was similar to that of the IDOL2 database. Image sequences were acquired using a mobile robot platform, under several illumination conditions (sunny, cloudy, night) and across several days. As in case of IDOL2, special care was taken to capture people's activities and change of location of objects and furniture (see Fig. 5). However, the acquisition was performed using both perspective and omnidirectional cameras, in several parts of a different environment and using different hardware. For further details, we refer the reader to [38].

For our experiments, we employed only the perspective images and we selected six different extended sequences from the database. The extended sequences were acquired in a larger section of the environment consisting of nine rooms of different functionality: a corridor, a printer area, a kitchen, a large office, 2 two-persons offices, a one-person office, a bathroom and a stairs area. The sequences contained on average 2547 frames. The six sequences were selected to mimic the organization of the IDOL2 database. For each illumination setting, we chose two sequences acquired under similar conditions and close in time.

5.3. Image descriptors

Two visual descriptors, global and local, were employed during our experiments. We used Composed Receptive Field Histograms (CRFH [26]) as global features. CRFHs are a multi-dimensional statistical representation of the occurrence of responses of several im-

Fig. 3. Robot platforms employed in the experiments with the IDOL2 database and images illustrating the appearance of the five rooms from the robots' the point of view.

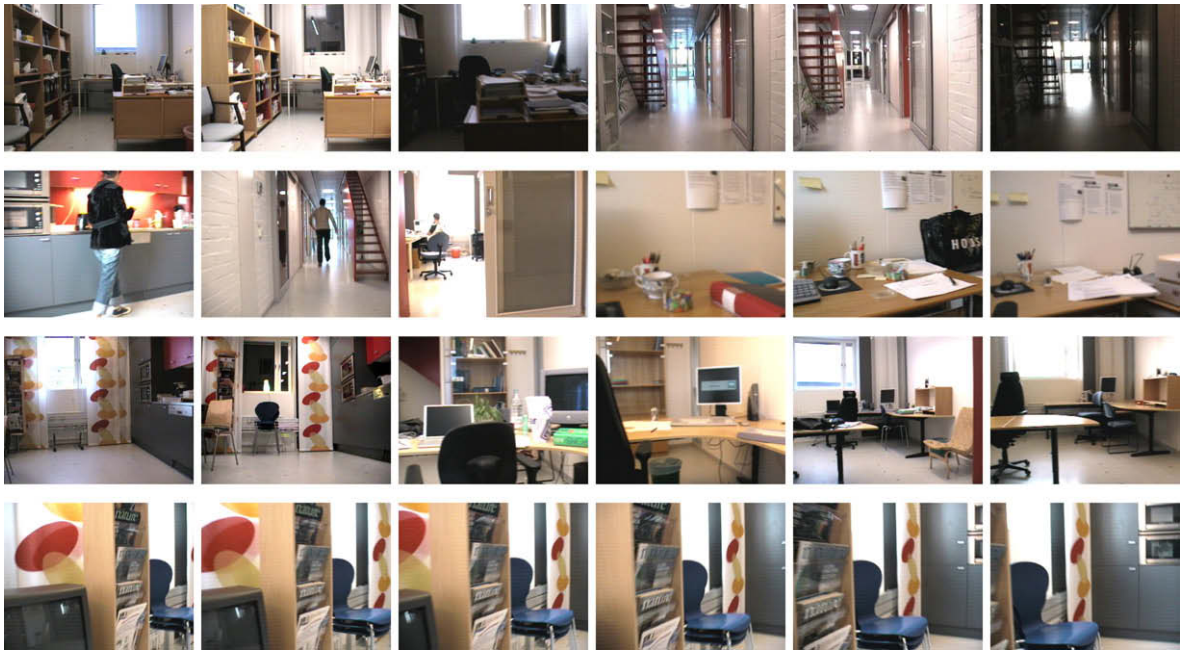


Fig. 4. Sample images illustrating the variations captured in the IDOL2 database. Images in the top row show the variability introduced by changes in illumination for two rooms. The second and third rows show people appearing in the environment (first three images, second row) as well as the influence of people's activity including some larger variations which happened over a time span of 6 months. Finally, the bottom row illustrates the changes in viewpoint observed for a series of images acquired one after another in 1.2 s.

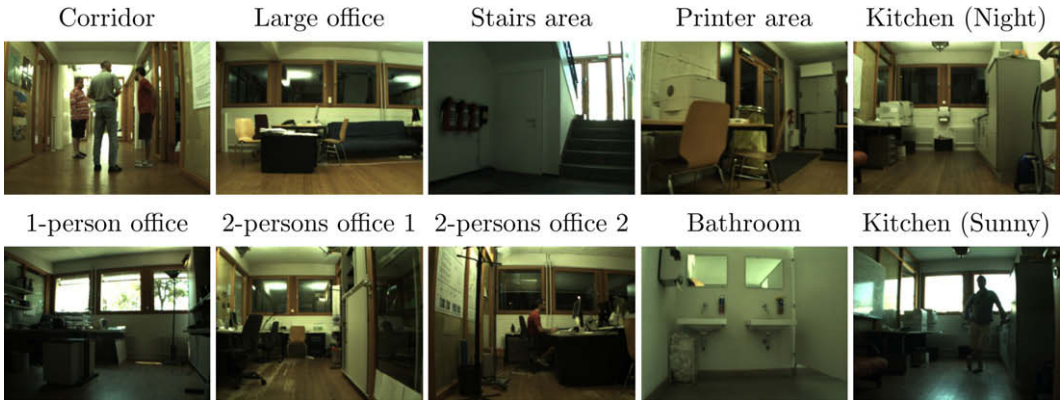


Fig. 5. Sample images from the COLD-Freiburg database illustrating the rooms in which acquisition was performed and different types of captured variability introduced by human activity and changes in illumination.

age descriptors applied to the image. This idea is illustrated in Fig. 6. Each dimension corresponds to one descriptor and the cells of the histogram count the pixels sharing similar responses of all descriptors. This approach allows to capture various properties of the image as well as relations that occur between them. Multi-dimensional histograms can be extremely memory consuming and computationally expensive if the number of dimensions grows. In [26], Linde and Lindeberg suggest to exploit the fact that most of the cells are usually empty, and to store only those that are non-zero. This representation allows not only to reduce the amount of memory required, but also to perform operations such as histogram accumulation and comparison efficiently.

The idea behind local features is to represent the appearance of an image only around a set of characteristic points known as the interest points. The similarity between two images is then measured by solving the correspondence problem. Local features are known to be robust to occlusions and viewpoint changes, as the absence of some interest points does not affect the features extracted from other local patches. The process of local feature extraction consists of two stages: *interest point detection* and *description*. The interest point detector identifies a set of characteristic points in the image that could be re-detected even in spite of various transformations (e.g. rotation and scaling) and variations in illumination conditions. The role of the descriptor is to extract robust features from the local patches located at the detected points. In this paper, we used the scale, rotation, and translation invariant Harris–Laplace detector [31] and the SIFT descriptor [27]. Fig. 7 shows two examples of interest point detected on images of indoor environments.

5.4. Parameter selection and performance evaluation

For all experiments, the kernel parameter and the SVM cost parameter C were determined via cross validation, separately for each database. Then, the obtained values were used as constants for all the incremental learning experiments. For all experiments,

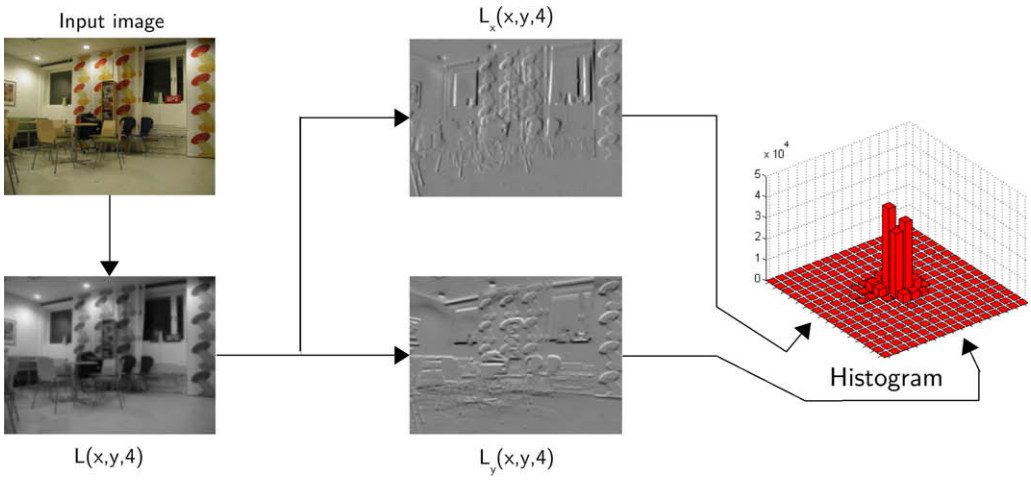


Fig. 6. The process of generating multi-dimensional receptive field histograms using the first-order derivatives computed at the scale $t = 4$ and the number of bins per dimension set to 16.

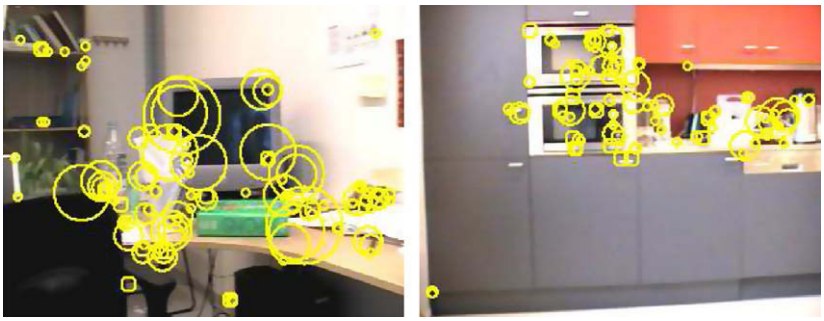


Fig. 7. Examples of images marked with interest points detected using the Harris–Laplace detector. The radius of the circles illustrate the scale at which the points were detected.

we used the implementation of SVM provided by the *libsvm* library [9].

Since the employed datasets are unbalanced (e.g. in case of the IDOL2 database there are on average 443 samples for CR, 114 for 1pO, 129 for 2pO, 133 for KT and 135 for PR), as a measure of performance for the reported results and parameter selection, we used the average of classification rates obtained separately for each actual class. For each single experiment, the percentage of properly classified samples was first calculated separately for each room and then averaged with equal weights independently of the number of samples acquired in the room. This allowed to eliminate the influence that large classes could have on the performance score.

In our experiments, we observed a few percent improvement of the final results when a performance measure that is not invariant to unbalanced classes was used. This was caused by very good performance of the system for the corridor class. The was visually distinct from the other classes and was represented by the largest number of samples. As a result, in our experiments, the measure was used mainly to compensate for the influence of the corridor class.

6. Experiments on support vector reduction

To begin with, we run some experiments to evaluate the behavior of the support vector reduction algorithm described in Section 4.3. We used two sequences from the IDOL2 database [28], one as train set and the other as test set. We chose CRFH as an image descriptor, and trained SVMs with four different types of kernels: linear kernel, RBF kernel, χ^2 kernel and histogram intersection (Hist.-Inte.) kernel. First, the SVM classifier was trained using the SMO algorithm. Then, starting from the obtained discriminative function, the reduction algorithm was tested, for different values of the reduction threshold τ. After each experiment (for each value of τ), the original model was reduced and the number of kept support vectors and the performance of the reduced model were tested on the same test set. If the classification rate dropped below 80% of the initial classification rate, i.e. $\Theta < 80\%$, the process was stopped. Fig. 8 reports the percentage of the reduced number of support vectors (SV) compared to the initial model (left), and the percentage of the initial classification rate that is preserved after the reduction (right), as a function of different value of τ. We see that, apart for the linear kernel, the algorithm behaves as expected, obtaining a gentle decrease in performance as the number of stored support vectors is being reduced. It is worth noting that the linear kernel is known for being not a good metric for histogram-like features, as instead all the other three kernels are. This might explain its different behavior.

7. Experiments on adaptation

As a first application of our method, we present experiments on visual place recognition in highly dynamic indoor environments. We consider a realistic scenario, where places change their visual appearance because of varying illumination conditions or human activity. Specifically, we focus on the ability of the recognition algorithm to adapt to these changes over long periods of time. As it is not possible to predict in advance the type of changes that will occur, adaptation must be performed incrementally.

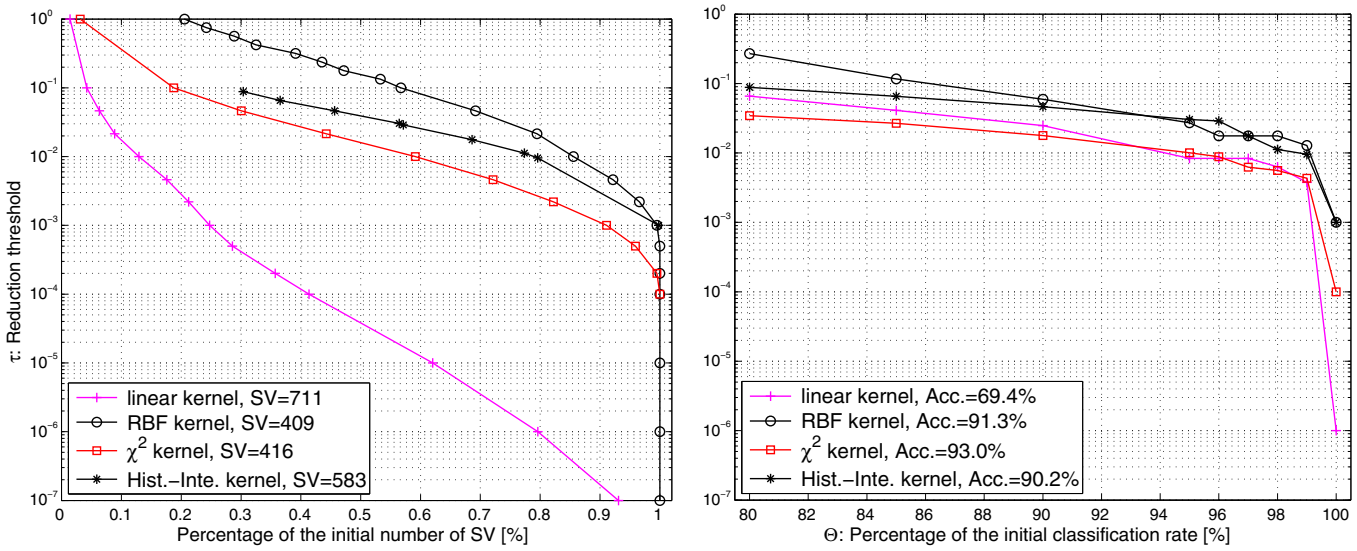


Fig. 8. Percentage of the reduced number of support vectors (SV) compared to the initial model (left), and the percentage of the original classification rate that is preserved after the reduction (right), both as a function of different value of τ for various kernel types. The initial number of support vectors (SV) and initial classification rate (Acc.) were reported for each kernel.

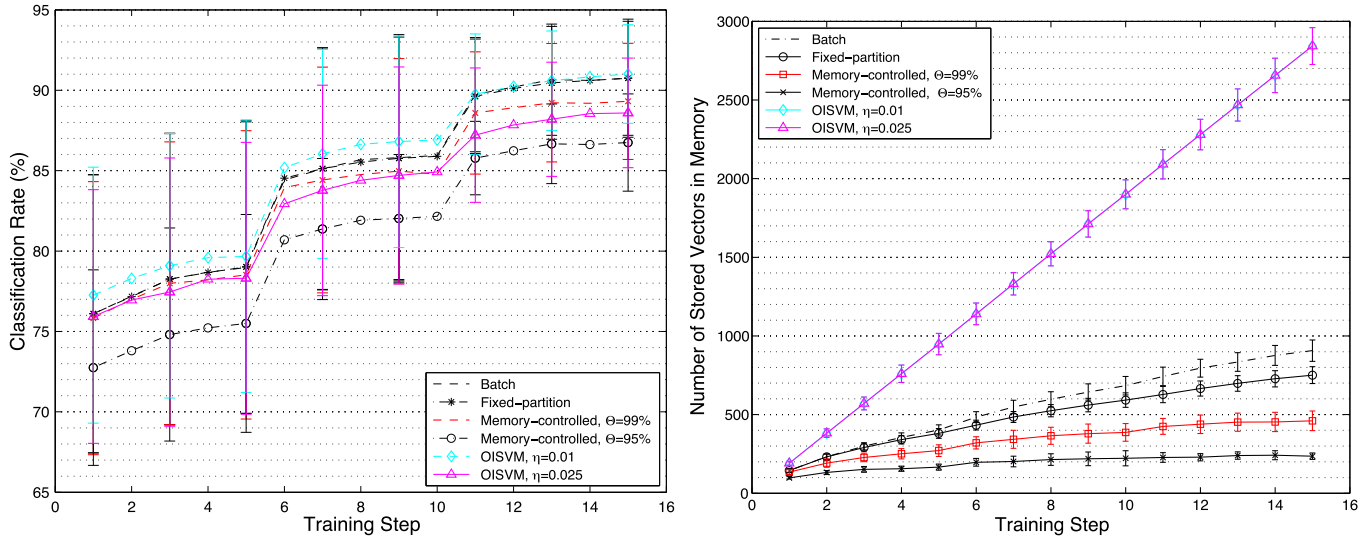
We conducted two series of experiments to evaluate the effectiveness of the memory-controlled incremental SVM for this task. In the first, we considered a case in which the variability observed by the recognition system was *constrained* to changes introduced by long-term human activity under stable illumination conditions. Such experimental procedure allowed us to thoroughly examine the properties of each of the incremental methods in a more controlled setting. The corresponding experiments are reported in Section 7.1. In the second, we considered a real-world, *unconstrained* scenario, where the algorithms had to incrementally gain robustness to variations introduced by changing illumination and short-term human activity, and then, to use their adaptation abilities to handle long-time environment changes. The corresponding experiments are reported in Section 7.2. In both experiments, we compared our approach with the fixed-partition incremental SVM, OISVM and the batch method. This last algorithm is used here purely as a reference, as it is not incremental. We used CRFH global image features. We tested a wide variety of combinations of image descriptors, with several scale levels [36]. On the basis of an evaluation of performance and computational cost, we built the histograms from normalized Gaussian derivative filters applied to the images at two different scales, and we used χ^2 as a kernel for SVM. We also performed experiments using SIFT local features combined with the matching kernel for SVM. Both types of features previously proved effective for the place recognition task [40,39].

7.1. Experiments with constrained variability

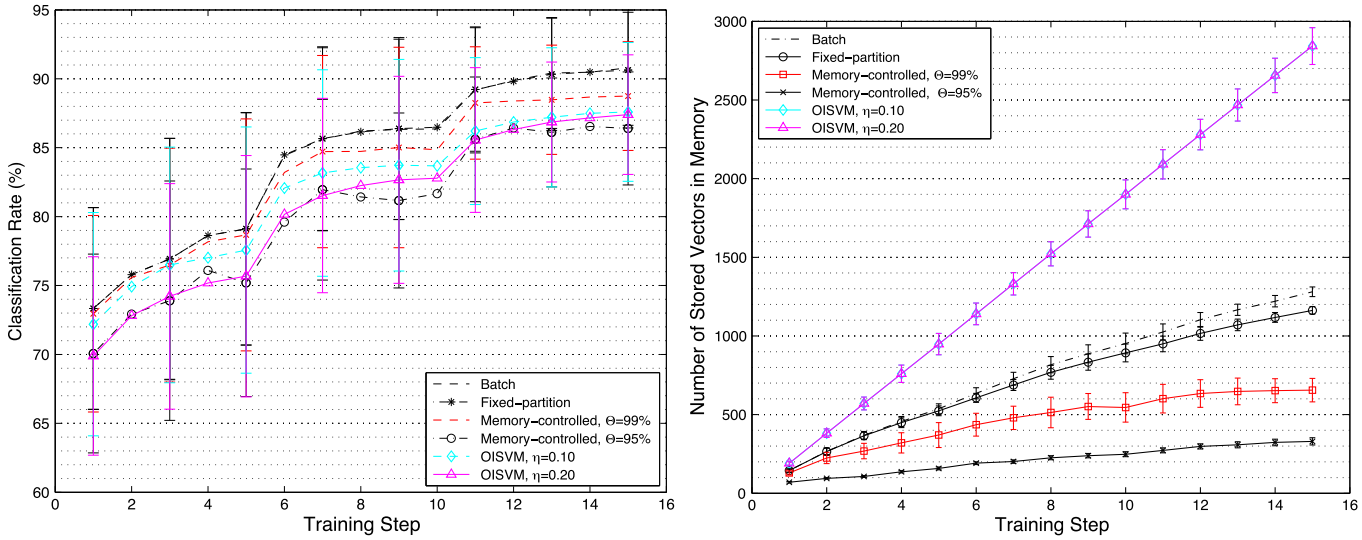
In the first series of experiments, we evaluated the properties of the memory-controlled incremental SVM in a simplified scenario. We therefore trained the system on three sequences acquired under similar illumination conditions, with the same robot platform. The fourth sequence was used for testing. Training on each sequence was performed in five steps, using one subsequence at a time, resulting in 15 steps in total. We considered 36 different permutations of training and test sequences. Here we report average results obtained on both global and local features by the three incremental algorithms (fixed-partition, OISVM, and memory-controlled) as well as the batch method. We tested the memory-controlled algorithm using two different values of the parameter Θ, i.e. $\Theta = 99\%, 95\%$. This corresponds to the maximum accepted reduction of the recognition rate of 1% and 5% respectively, as explained in Section 4.3. Similarly for OISVM, we used three different values of the parameter η that determines how sparse the final solution is going to be (as in [35]).

Fig. 9, left, shows the recognition rates obtained at each incremental step by all methods and for both feature types. Fig. 9, right, reports the number of training samples that had to be stored in the memory at each step of the incremental procedure. First, we see that OISVM achieves very good performance similar to the batch method. However, both methods suffer from the same problem: they require all the training samples to be kept in the memory during the whole learning process. This makes them unsuitable for realistic scenarios, particularly in cases when the algorithm should be used on a robotic platform with intrinsically limited resources. The fixed-partition algorithm achieves identical performance as the batch method, while greatly reducing the number of training samples that need to be stored in the memory at each incremental step. However, despite that all the algorithms show plateaus in the classification rate whenever the model is trained on similar data (coming from consecutive subsequences), the number of support vectors grows roughly linearly with the number of training steps.

We see that for the memory-controlled incremental SVM, both the classification rate and the number of stored support vectors show plateaus every five incremental steps (as opposed to the classification rate only in case of the other methods). The method controls the memory growth much more successfully than the original fixed-partition incremental technique. For instance, when we accept only one percent reduction in classification (i.e. $\Theta = 99\%$), the number of support vectors stored after the 15 steps is 39.6% (CRFH) and 43.7% (SIFT) lower than for the fixed-partition incremental method. For $\Theta = 95\%$, the gain in memory compression is much greater than the overall decrease in performance. This feature, i.e. the possibility to trade memory for a controlled reduction in performance, can be potentially very useful for systems operating in realistic, open-ended learning scenarios and with limited memory resources. This approach would be even more appealing for systems which can compensate the loss in performance by doing information fusion over time or from multiple sensors. It is worth underlying that the growth in the number of support vectors decreases over time (Fig. 9, bottom). For example, for CRFH and $\Theta = 99\%$, the model trained on the second sequence (steps 6–10)



(a) Classification rate and number of training samples stored for global features.



(b) Classification rate and number of training samples stored for local features.

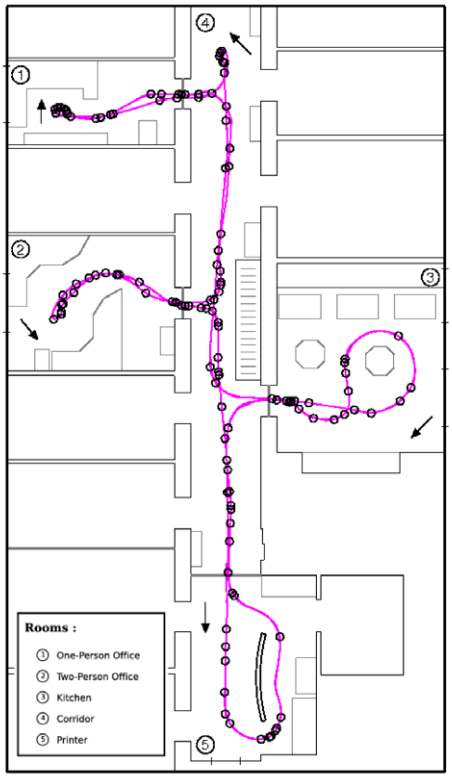
Fig. 9. Average results obtained for the experiments with constrained variability for three incremental methods and the batch algorithm.

grows by 115 vectors on average, but trained on the third sequence (steps 11–15) grows only by 74 vectors. This may indicate that the number of SVs eventually tends to reach a plateau.

In order to gain a better understanding of the methods' behavior, we performed an additional analysis of the results. Fig. 11b shows, for the two approximate incremental techniques, the average amounts of vectors (originating from each of the three training sequences) that remained in the model after the final incremental step (note that, in our case, this analysis would be pointless for OISVM, as it requires storing all the training data). The figure illustrates how the methods weigh instances, learned at different time, when constructing the internal representation. We see that both fixed-partition and memory-controlled algorithms privilege new data, as the SVs from the last training sequence are more represented in the model. This phenomenon is stronger for the memory-controlled algorithm.

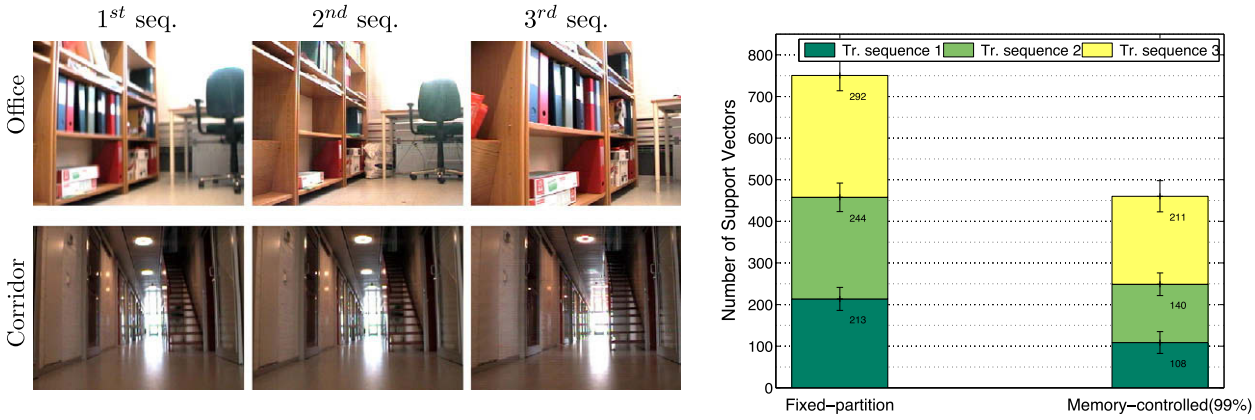
To get a feeling for how the forgetting capability works in case of the memory-controlled method, we plotted the positions where the SVs were acquired, for $\Theta = 99\%$ and the CRFH features. Fig. 10 reports results obtained for a model built after the final incremental step. The positions were marked on three maps presented in Fig. 10a–c so that each of the maps shows the SVs originating from only one training sequence. These SVs could be considered as landmarks selected by the visual system for the recognition task. As already shown in Fig. 11b, most of the vectors in the model come from the last training sequence. Moreover, the number of SVs from the previous training steps decreases monotonically, thus the algorithm gradually forgets the old knowledge. It is interesting to observe how the vectors from each sequence are distributed along the path of the robot. On each map, the places crowded with SVs are mainly transition areas between the rooms, regions of high variability, as well as places at which the robot rotated (thus providing a lot of different visual cues without changing position). To illustrate the point, Fig. 11a shows sample images acquired in the corridor, for which the SVs decay quickly, and one of the offices, for which they are being preserved much longer. The results indicate that the forgetting is not performed randomly. On the contrary, the algorithm tends to preserve those training vectors that are

(a) 78 Support Vectors from 1st seq. (b) 111 Support Vectors from 2nd seq.



(c) 149 Support Vectors from 3rd seq.

Fig. 10. Maps of the environment with plotted positions of the support vectors stored in the model obtained after the final incremental step for one of the experiments conducted using the memory-controlled technique with $\Theta = 99\%$. The support vectors were divided into three maps (a–c) according to the training sequence they originate from. Additionally, each map shows the path of the robot during acquisition of the sequence (arrows indicate the direction of driving). We observe that the support vectors from the old training sequences were gradually eliminated by the algorithm and this effect was stronger in regions with lower variability.



(a) Sample images from the three training sequences.

(b) Statistics of Support Vectors stored in the final approximate incremental models.

Fig. 11. Sample images captured in regions of different variability (left). Comparison of the average amounts of training vectors coming from the three sequences that were stored in the final incremental model for the two approximate incremental techniques (right).

most crucial for discriminative classification, and first forgets the most redundant ones.

On the basis of these experimental findings, we can conclude that the memory-controlled incremental SVM is the best method for vision-based robot localization of those considered here. Therefore, in the rest of the paper we will use only this algorithm, with $\Theta = 99\%$.

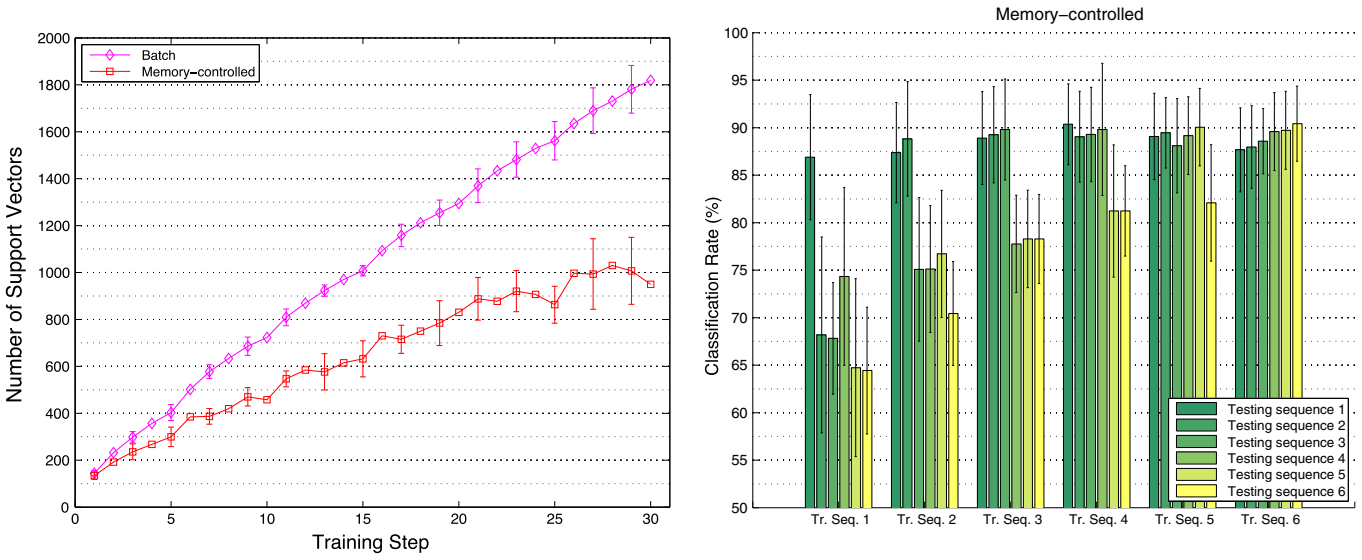
7.2. Experiments with unconstrained variability

The next step was to test our incremental method in a real-world scenario. To this purpose, we considered the case where the algorithm needed to incrementally gain robustness to variations introduced by changing illumination and human activities, while at the same time using its adaptation ability to handle long-time changes in the environment. We performed the experiments first on the IDOL2 database. Then, to confirm the behavior on a different set of data, we used the COLD-Freiburg database. We first trained the system on three IDOL2 sequences acquired at roughly similar time but under different illumination conditions. Then, we repeated the same training procedure on sequences acquired 6 months later. In order to increase the number of incremental steps and differentiate the amount of new information introduced by each set of data, each sequence was again divided into five subsequences. In total, for each experiment we performed 30 incremental steps. Since the IDOL2 database consists of pairs of sequences acquired under roughly similar conditions, each training sequence has a corresponding one which could be used for testing. Feature-wise, here we used only the global features (CRFH). Indeed, the experiments presented in the previous section showed that local features achieve an accuracy similar to that of CRFH, but at a much higher computational cost and memory requirement. Also, preliminary experiments show that this behavior is confirmed in this scenario, hence the choice to use here only the global descriptor.

We used a very similar system and experimental procedure for the experiments with the COLD-Freiburg dataset. As in case of IDOL2, we divided each sequence into five subsequences and used pairs of sequences acquired under roughly similar conditions for training and testing. In case of both databases, the experiment was repeated 12 times for different orderings of training sequences. Figs. 12 and 13 report the average results together with standard deviations. By observing the classification rates for a classifier trained on the first sequence only, we see that the system achieves best performance on a test set acquired under similar conditions. The classification rate is significantly lower for other test sets. In case of IDOL2, this is especially visible for images acquired 6 months later, even under similar illumination conditions. At the same time, the performance greatly improves when incremental learning is performed on new batches of data. The classification rate decreases for the old test sets; at the same time, the size of the model tends to stabilize.

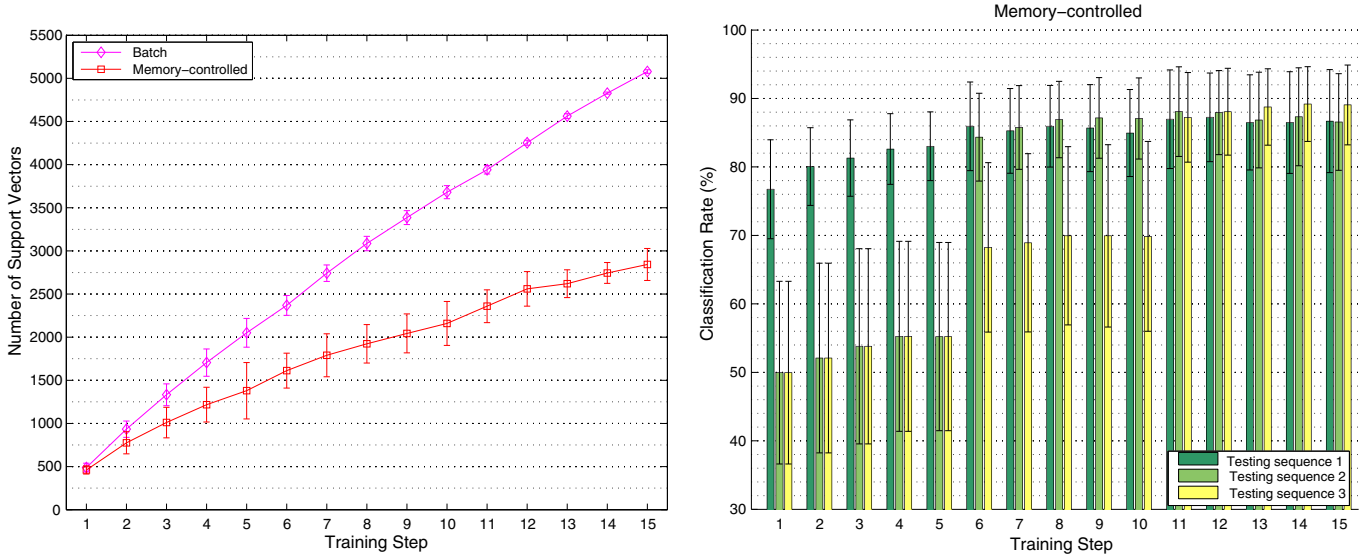
7.3. Discussion

The presented results provide a clear evidence of the capability of the discriminative methods to perform incremental learning for vision-based place recognition, and their adaptability to variations in the environment. Table 1 summarizes the performance obtained by each method in terms of accuracy, speed, controlled memory growth and forgetting capability. For each algorithm (i.e. for each row), we put a cross corresponding to the property (i.e. the column) that the algorithm has shown to possess in our experiments. The fixed-partition method performs as well as batch SVM, but it is unable to control the memory growth and requires much more memory space. We also found that OISVM could get very good accuracy while achieving a low computational complexity during testing. However, none of the two methods has shown to possess an effective forgetting capability: for the fixed-partition method, the old SVs decay slowly, but the decay is neither predictable nor controllable; for OISVM, every training vector must be stored into memory. As opposed to this, the memory-controlled algorithm is able to achieve performances statistically equivalent to those of batch SVM, while at the same time providing a principled and effective way to control the memory growth. Experiments showed that this has induced a forgetting capability which privileges newly acquired data to the expenses of old one and the model growth slows down whenever new data are similar to those already processed. Furthermore, since a lot of training images can be discarded during the incremental process, the training time soon becomes significantly lower than for the batch method. For instance, in case of the second experiments, training the classifier at the last step took 25.5 s for the batch algorithm and only 5.6 s for the memory-controlled method on a 2.6 GHZ Pentium IV machine, and recognition time was twice as fast for the memory-controlled algorithm than for the batch one.



(a) Support vectors (b) Performance of memory-reduced

Fig. 12. Average results of the IDOL2 experiments in the real-world scenario. (a) Compares the amounts of SVs stored in the models at each incremental step for the batch and the memory-controlled method. (b) Reports the classification rate measured every fifth step (every time the system completes learning a whole sequence) with all the available test sets. The training and test sets marked with the same indices were acquired under similar conditions.



(a) Support vectors (b) Performance of memory-reduced

Fig. 13. Average results of the COLD-Freiburg experiments in the real-world scenario. (a) Compares the amounts of SVs stored in the models at each incremental step for the batch and the memory-controlled method. (b) Reports the classification rate measured every step with all the available test sets. The consecutive training and testing sequences were acquired under similar conditions.

Table 1
Comparing incremental learning techniques for place recognition and robot localization applications.

	Accuracy	Forgetting	Memory	Speed
Fixed-partition	x	x		
OISVM	x			x
Memory-controlled	**x**	**x**	**x**	**x**

8. Experiments on knowledge transfer

As a second application of our method, we considered the problem of transfer of knowledge between robotic platforms with different characteristics, performing vision-based recognition in the same environment. We used the IDOL2 database and the robots Minnie and Dumbo for these experiments. The main difference between the two platforms lies in the height of the cameras (see Fig. 15). They both use the memory-controlled incremental SVM as a basis for their recognition system, thus they share the same knowledge representation. The aim is to efficiently exploit the knowledge acquired e.g. by one robot so to boost the recognition performance of another robot. We propose to use our method to update the internal representation when new training data are available. Fig. 14 illustrates how our approach can be used for transfer of knowledge. We would like the knowledge transfer scheme to be adaptive, and also to privilege newest data so to

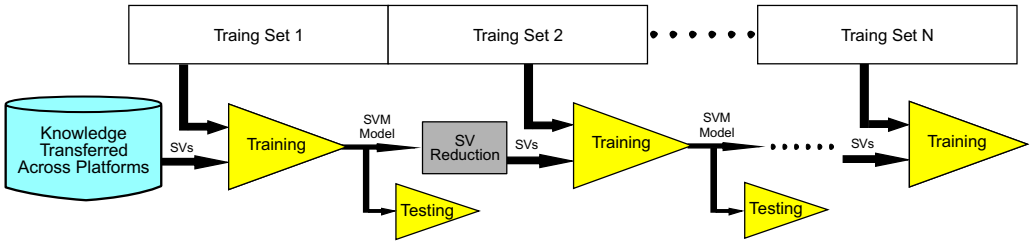


Fig. 14. A diagram illustrating the data flow in the knowledge-transfer system.

avoid accumulation of outdated information. Finally, the solution obtained starting from a transferred model should gradually converge to the one learned from scratch, not only in terms of performance but also of required resources (e.g. memory).

The challenges in the transfer of knowledge will come from:

- *(a) Differences in the parameters of the two platforms*: The cameras are mounted at two different heights, thus the informative content of the images acquired by the two platforms is different. Because of this, the knowledge acquired by one platform might not be helpful for the other one or, in the worst case, it might constitute an obstacle. Preliminary experiments showed that SIFT is more suitable for the transfer of knowledge in our scenario than CRFH. For that reason, CRFH will not be used.
- *(b) Room by room/frames by frames knowledge update*: It is desirable to update the model transferred across platforms as soon as new data are available. We will investigate the behavior of the algorithm when the update is performed room by room, or frames by frames. Both scenarios are at risk of unbalanced data with respect to the class being updated.
- *(c) Growing memory requirements*: Building on top of an already trained classifier might lead to a solution that will be much more demanding in terms of memory usage and computational power than the one learned from scratch. Although our memory-controlled approach is capable of reducing the number of SVs, its reduction process does not take the sources of the information into consideration. In order to favor information coming from the platform currently in use, we imposed to the algorithm to discard only those SVs that were linearly dependent *and* came from the previous platform by adding meta-information on the training examples. This scheme speeds up the turnover of stored SVs, while preferring newest data and at the same time preserving relevant information.

In the IDOL2 database, for each robot and for every illumination condition, we always have two sequences acquired under similar conditions. Here, we always used such pairs of sequences, one as a training set and the other one as a test set. In all the experiments, we benchmarked against a system not using any prior knowledge.

8.1. Experiments with room by room updates

In the first series of experiments, the system was updated incrementally in a room by room (i.e. class by class) scenario. The system was trained incrementally on one sequence; the corresponding sequence, acquired under roughly similar conditions, was used for testing. The prior-knowledge model was built using standard batch SVM from one image sequence, acquired under the same illumination conditions and at close time as the training one, but using a different platform. As there are five classes in total, training was performed in five steps (the algorithm learned incrementally one room at the time). In the no-transfer case, the system needed to build the model from scratch, and thus needed to acquire data from at least two classes. In this case, training on each sequence required only four steps since in the first step the algorithm learned to distinguish between the first two classes.

Building on top of knowledge acquired from another platform implies a growth in the memory requirements. To evaluate this behavior in relationship to its effects on performance and compare fairly to the system trained without a prior model, we incrementally updated the model without transferred knowledge on another sequence acquired under conditions similar to that of the first training sequence. This experiment makes it possible to evaluate performance and memory growth when both systems are trained on two sequences. The main difference is that in one case both sequences were acquired and processed by the same platform; in the other case, one sequence was acquired and processed by a different

Fig. 15. Knowledge transfer across robot platforms which only partially share visual information.

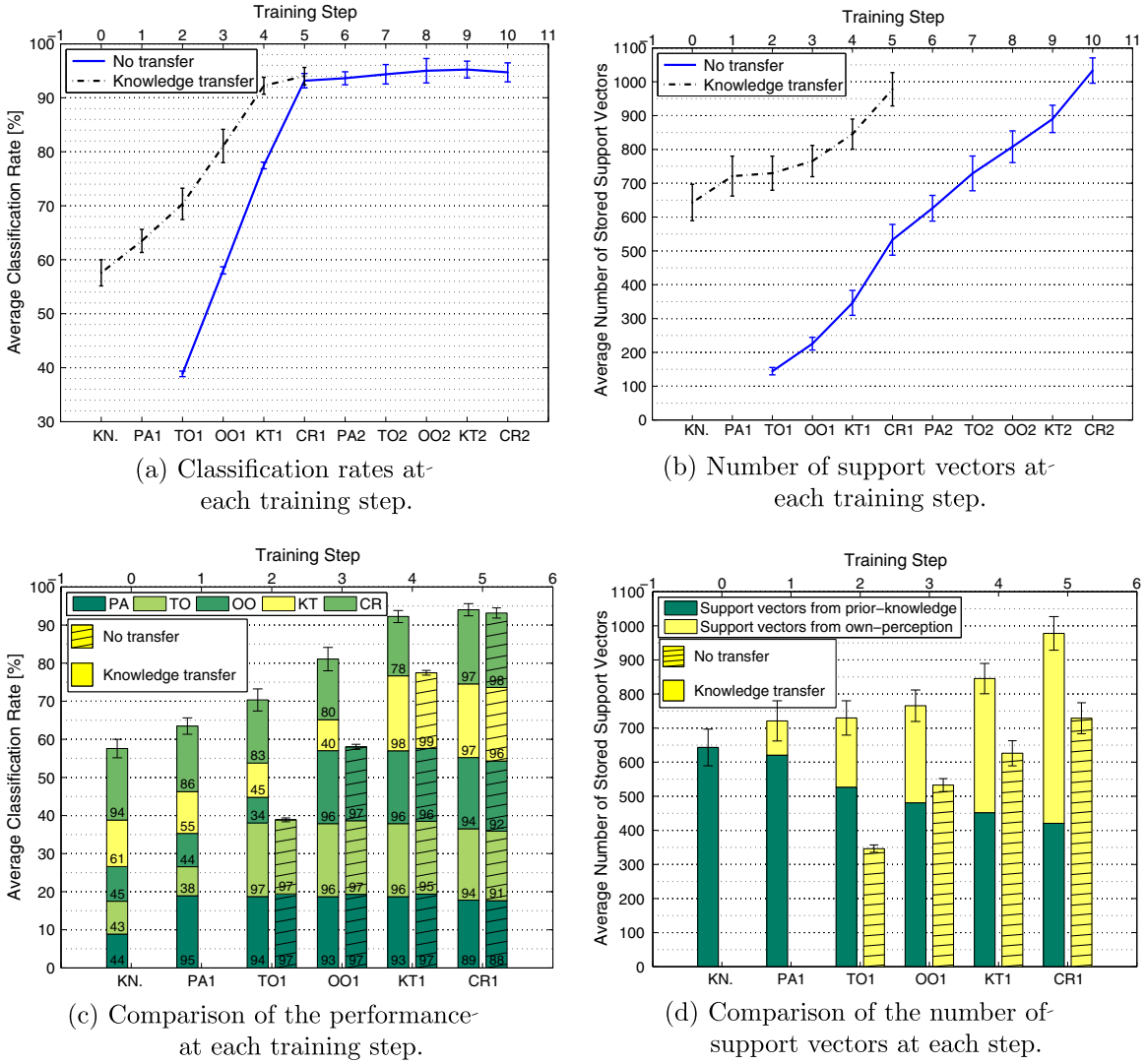


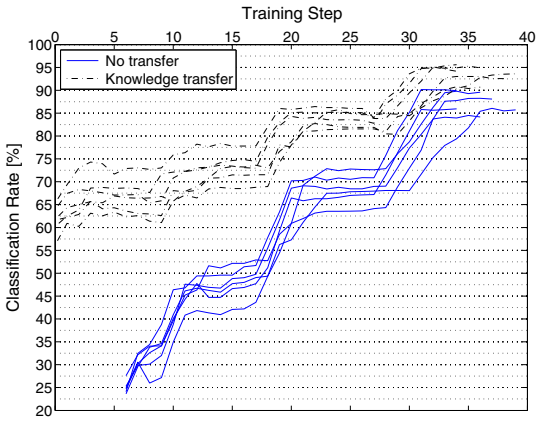
Fig. 16. Average results obtained for the system incrementally trained with and without transfer of knowledge in the room by room fashion. (a and b) Compares the final recognition rates and the total number of support vectors for both cases. (c and d) Presents a detailed analysis: classification rates obtained for each of the rooms and the amount of support vectors in the final model that originate from the transferred knowledge. In all the plots, the first step "KN" corresponds to the results obtained for the transferred knowledge before any update was performed.

platform. We considered different permutations in the rooms order for the updating; for each permutation, we considered six different orderings of the sequences used as training, testing, and prior-knowledge sets. Due to space reasons, we report only average results for one permutation, together with standard deviations in Fig. 16.

We can see that, for both approaches, the system gradually adapts to its own perception of the environment. It is clear that the knowledge-transfer system has a great advantage in terms of performance over the no-transfer system at the first steps. For instance, we see that, after the second update (TO1, Fig. 16a), the knowledge-transfer system achieves a classification rate of 65.3%, while the no-transfer knowledge obtains only 37%. The advantage in classification rate for the knowledge-transfer system remains considerable for the steps OO1 and KT1. However, it is interesting to note that even when both systems have been updated on a full sequence (CR1, Fig. 16a), the knowledge-transfer system still maintains an advantage in performance. Considering the differences between the two platforms, and that the transferred knowledge model was built on a single sequence, this is a remarkable result. It can also be observed from Fig. 16d that the memory-controlled algorithm facilitated the decay of knowledge from the other platform (in the first incremental step, we did not perform the reduction), while the knowledge acquired by its own sensor gradually becomes the main source for the model. As the no-transfer system continued to learn one additional sequence incrementally, its memory growth eventually exceeded the knowledge-transfer case (see Fig. 16b). Although the model was built on two sequences acquired by the same platform, the knowledge-transfer system still obtains a comparable performance. We conclude that the transfer of knowledge, in a room by room updating scenario, acts as an effective boosting of performance, without any long-term growth of the memory requirements.

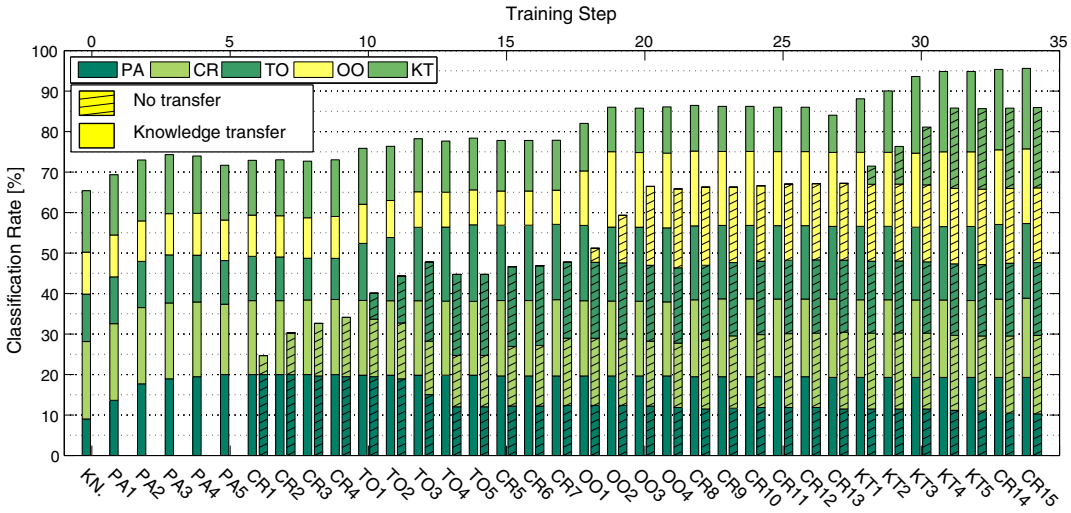
8.2. Experiments with frame by frame updates

The second series of experiments explored the behavior of the system in a frames by frames updating scenario. Here, for each incremental update, we used a certain number of consecutive frames taken from the training image sequence. Again, the system was trained incrementally on one sequence, and a corresponding sequence was used as a test set. We examined the performance

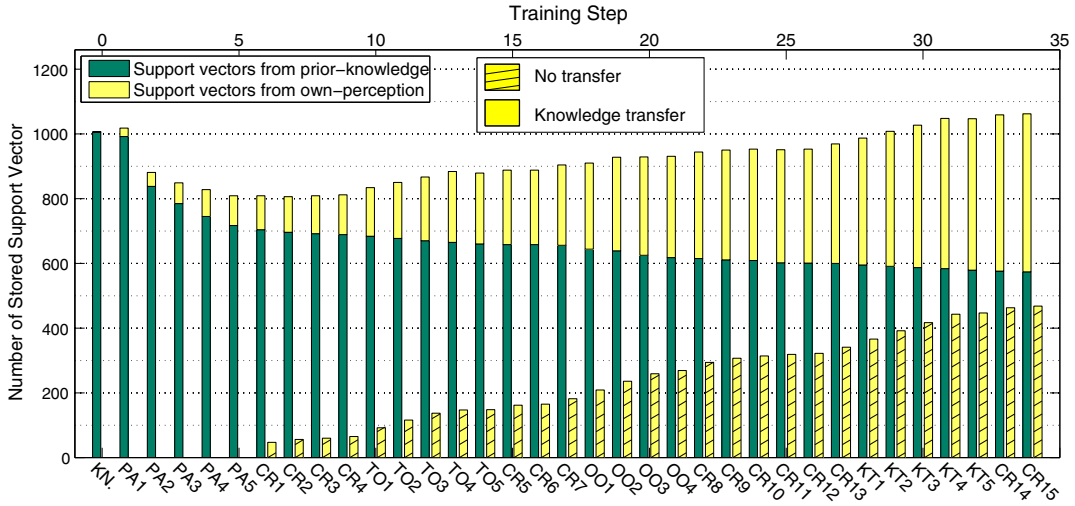


(a) Classification rates at each training step.

(b) Number of support vectors at each training step.



(c) Detailed comparison of the performance of the system with and without knowledge-transfer.



(d) Number of stored support vectors of incremental experiment with and without knowledge-transfer at each step.

Fig. 17. Average results obtained for the system incrementally trained with and without transfer of knowledge in the frames by frames fashion. The labels below each bar indicate the batch of data used for the incremental update. Again, the first step labeled as "KN" corresponds to the results obtained for the transferred knowledge before any update was performed.

of the system for the case when updating was performed using 30 frames per step.[1] Thus, for each experiment, it took more than 30 incremental steps in total to complete a sequence. The prior-knowledge model was built using two complete sequences acquired by the other platform, under the same illumination conditions and very close in time. This provided a better start-up performance than in case of the previous experiments. Again, we benchmarked against the system not using any prior knowledge. In this case, in order to fulfill the requirement of training using at least two classes, the first training set consisted of all the images captured in the first room plus the first 30 frames captured in the second room. As a consequence, the full training process required five to six less steps than in case of equivalent experiments using the knowledge-transfer scheme. The experiment was repeated six times for different orderings of training sequences. Since the number of training steps varied (due to a different number of images in each sequence), we report all the results separately. Fig. 17a and b reports the amount of stored SVs and classification rates at each step, for all the experiments. This shows the general behavior for both approaches. Fig. 17c and d presents results for one of the six experiments, so to allow a detailed analysis.

By observing the classification rates obtained at each step in both cases, we see that the advantage of the knowledge-transfer scheme is even more visible here than for the room by room updating scenario. This might be due to the fact that some of the training sets used for the no-transfer case are highly unbalanced. We can observe from Fig. 17c that the performance of the system for previously learned rooms can drop considerably when a new batch of frames is loaded; this is not the case for the knowledge-transfer system. The twelfth step, when the system was updated with frames from the two-persons office (TO3, Fig. 17c), is a typical example. Note that this is a general phenomenon present, although less pronounced, also in the room by room updating scenario. Our interpretation is that the model of the prior-knowledge contains information about the overall distribution of the data. This helps to find a balanced solution when dealing with non-separable instances using soft-margin SVM [10]. As a last remark the knowledge from the transferred model is gradually removed over time (see Fig. 17d).

9. Summary and conclusions

In this paper we presented a novel extension of SVM to incremental learning that achieves the same recognition performance of the standard, batch method while limiting the memory growth over time. This is achieved by discarding, at each incremental step, all the support vectors that are not linearly independent. The information they carry is not lost, as it is retained into the algorithm's decision function in the form of weighting coefficients of the remaining support vectors. We call this method memory-controlled incremental SVM. We applied it to the problem of place recognition for robot topological localization, focusing on two distinct scenarios: adaptation in presence of dynamic changes and transfer of knowledge between two robot platforms engaged in the same task. Experiments show clearly the effectiveness of our approach in terms of accuracy, speed, reduced memory and capability to forget redundant, outdated information.

We plan to extend this work in several ways. First, we want to use the memory-controlled algorithm in multi-modal learning scenarios, for instance using laser-based features combined with visual ones, as done in [40], in an incremental setting. Here we should be able to exploit fully the properties of the method, and aggressively trade memory for accuracy on single modalities, while retaining an high overall performance. Second, we would like to investigate further the knowledge transfer scenario, and incorporate in our framework ways to select the data to be transferred, as proposed in [25]. Future work will concentrate in these directions.

[1] Experiments conducted for 10 and 50 frames per training step gave analogous results, and for space reasons are not reported here.

Acknowledgments

This work was sponsored by the EU FP7 Project CogX (A. Pronobis) and IST-027787 DIRAC (B. Caputo, L. Jie), and the Swedish Research Council Contract 2005-3600-Complex (A. Pronobis). The support is gratefully acknowledged.

References

[1] COGNIRON: The Cognitive Robot Companion, <http://www.cogniron.org>.
[2] CoSy: Cognitive Systems for Cognitive Assistants, <http://www.cognitivesystems.org/>.
[3] RobotCub, <http://www.robotcub.org/>.
[4] M. Artač, M. Jogan, A. Leonardis, Mobile robot localization using an incremental eigenspace model, in: Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA'02), 2002, pp. 1025–1030.
[5] S. Belongie, C. Fowlkes, F. Chun, J. Malik, Spectral partitioning with indefinite kernels using the nyström extension, in: Proceedings of the 7th European Conference on Computer Vision (ECCV'02), 2002, pp. 531–542.
[6] Emma Brunskill, Thomas Kollar, Nicholas Roy, Topological mapping using spectral clustering and classification, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Diego, October 2007.
[7] B. Caputo, E. Hayman, P. Mallikarjuna, Class-specific material categorisation, in: Proceedings of the 10th International Conference on Computer Vision (ICCV'05), 2005, pp. 1597–1604.
[8] G. Cauwenberghs, T. Poggio, Incremental and decremental support vector machine learning, in: Advances in Neural Information Processing Systems (NIPS), 13 (2001).
[9] Chih Chung Chang, Chih Jen Lin, LIBSVM: A Library for Support Vector Machines, 2001, <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
[10] N. Cristianini, J.S. Taylor, An Introduction to Support Vector Machines and other Kernel-based Learning Methods, Cambridge University Press, 2000.
[11] D. Skočaj, A. Leonardis, Weighted and robust incremental method for subspace learning, in: Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV'03), 2003, pp. 1494–1501.
[12] M. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba, A solution to the simultaneous localization and map building (SLAM) problem, IEEE Transactions on Robotics and Automation 17 (3) (2001) 229–241.
[13] C. Domeniconi, D. Gunopulos, Incremental support vector machine construction, in: Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM'01), 2001, pp. 589–592.
[14] Gyuri Dorkó, Cordelia Schmid, Object Class Recognition using Discriminative Local Features, 2005.
[15] Tom Downs, Kevin E. Gates, Annette Masters, Exact simplification of support vector solutions, The Journal of Machine Learning Research 2 (2002).
[16] J. Folkesson, P. Jensfelt, H. Christensen, Vision SLAM in the measurement subspace, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'05), Barcelona, Spain, 2005, pp. 30–35.
[17] M. Fritz, B. Leibe, B. Caputo, B. Schiele, Integrating representative and discriminant models for object category detection, in: Proceedings of the International Conference on Computer Vision (ICCV'05), 2005.
[18] G.H. Golub, C.F. Van Loan, Matrix Computations, third ed., Johns Hopkins University Press, 1996.
[19] M. Jogan, A. Leonardis, Robust localization using an omnidirectional appearance-based subspace model of environment, Robotics and Autonomous Systems 45 (1) (2003) 51–72.
[20] K. Grauman, T. Darrell, The pyramid match kernel: discriminative classification with sets of image features, in: Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV'05), 2005, pp. 1458–1465.
[21] G. Konidaris, A. Barto, Autonomous shaping: knowledge transfer in reinforcement learning, in: Proceedings of the 23rd International Conference on Machine Learning (ICML'06), 2006.
[22] D. Kortenkamp, T. Weymouth, Topological mapping for mobile robots using a combination of sonar and vision sensing, in: Proceedings of the 12th National Conference on Artificial Intelligence, Seattle, Washington, USA, 1994.
[23] Geert-Jan M. Kruijff, Hendrik Zender, Patric Jensfelt, Henrik I. Christensen, Situated dialogue and spatial organization: what, where...and why?, International Journal of Advanced Robotic Systems (ARS) 4 (1) (2007) 125–138 (special issue on human–robot interaction)
[24] B. Kuipers, P. Beeson, Bootstrap learning for place recognition, in: Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02), 2002.

[25] A. Lazaric, M. Restelli, A. Bonarini, Transfer of samples in batch reinforcement learning, in: Proceedings of the 25th International Conference on Machine Learning (ICML'08), Helsinki, Finland, 2008, pp. 544–551.
[26] O. Linde, T. Lindeberg, Object recognition using composed receptive field histograms of higher dimensionality, in: Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04), Cambridge, UK, 2004.
[27] D.G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.
[28] J. Luo, A. Pronobis, B. Caputo, P. Jensfelt, The IDOL2 Database. Technical Report 304, CVAP, KTH, 2006, <http://cogvis.nada.kth.se/IDOL2/>.
[29] R.J. Malak, Jr., P.K. Khosla, A framework for the adaptive transfer of robot skill knowledge using reinforcement learning agents, in: Proceedings of the 2001 IEEE International Conference on Robotics and Automation (ICRA'01), Seoul, Korea, 2001.
[30] O. Martínez Mozos, R. Triebel, P. Jensfelt, A. Rottmann, W. Burgard, Supervised semantic labeling of places using information extracted from sensor data, Robotics and Autonomous Systems 55 (5) (2007).
[31] K. Mikolajczyk, C. Schmid, Indexing based on scale invariant interest points, in: Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV'01), Vancouver, Canada, 2001.
[32] Tom Mitchell, The Discipline of Machine Learning, Technical Report CMU-ML-06-108, CMU, 2006.
[33] A.C. Murillo, J. Kosecka, J.J. Guerrero, C. Sagues, Visual door detection integrating appearance and shape cues, Robotics and Autonomous Systems 56 (6) (2008) 512–521.
[34] Illah Nourbakhsh, Rob Powers, Stan Birchfield, Dervish: an office navigation robot, AI Magazine 16 (2) (1995) 53–60.
[35] F. Orabona, C. Castellini, B. Caputo, J. Luo, G. Sandini, Indoor place recognition using online independent support vector machines, in: 18th British Machine Vision Conference (BMVC07), Warwick, UK, September 2007.
[36] A. Pronobis, Indoor Place Recognition Using Support Vector Machines, Master's Thesis, NADA/CVAP, Kungliga Tekniska Hoegskolan, Stockholm, Sweden, December 2005.
[37] A. Pronobis, B. Caputo, Confidence-based cue integration for visual place recognition, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07), San Diego, CA, USA, October 2007.
[38] A. Pronobis, B. Caputo, COLD: COsy localization database, The International Journal of Robotics Research (IJRR) 28 (5) (2009).
[39] A. Pronobis, B. Caputo, P. Jensfelt, H.I. Christensen, A discriminative approach to robust visual place recognition, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'06), Beijing, China, October 2006.
[40] A. Pronobis, O. Martínez Mozos, B. Caputo, SVM-based discriminative accumulation scheme for place recognition, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'08), Pasadena, CA, USA, May 2008.
[41] S. Se, D.G. Lowe, J. Little, Vision-based mobile robot localization and mapping using scale-invariant features, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'01), 2001, Seoul, Korea.
[42] Christian Siagian, Laurent Itti, Biologically-inspired robotics vision monte-carlo localization in the outdoor environment, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'07), San Diego, CA, USA, October 2007.
[43] N.A. Syed, H. Liu, K.K. Sung, Incremental learning with support vector machines, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'99), 1999.
[44] A. Tapus, R. Siegwart, Incremental robot mapping with fingerprints of places, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05), Edmonton, Alberta, Canada, August 2005.
[45] Sebastian Thrun, Learning metric-topological maps for indoor mobile robot navigation, Artificial Intelligence 1999 (1) (1998).
[46] Sebastian Thrun, Tom Mitchell, Lifelong robot learning, Robotics and Autonomous Systems 15 (1995).
[47] A. Torralba, K.P. Murphy, W.T. Freeman, M.A. Rubin, Context-based vision system for place and object recognition, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV'03), 2003, Nice, France.
[48] I. Ulrich, I. Nourbakhsh, Appearance-based place recognition for topological localization, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'00), San Francisco, CA, USA, 2000.
[49] Christoffer Valgren, Achim J. Lilienthal, SIFT, SURF and seasons: long-term outdoor localization using local features, in: Proceedings of the European Conference on Mobile Robots (ECMR'07), 2007.
[50] V. Vapnik, Statistical Learning Theory, Wiley and Son, 1998.
[51] C. Wallraven, B. Caputo, A. Graf, Recognition with local features: the kernel recipe, in: Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV'03), 2003.
[52] J. Wolf, W. Burgard, H. Burkhardt, Robust vision-based localization by combining an image retrieval system with Monte Carlo localization, IEEE Transactions on Robotics 21 (2) (2005) 208–216.