In *IEEE International Conference on Robotics and Automation* (ICRA 11),
Shanghai, China, May 9-13 2011.

# Autonomous Learning of Vision-based Layered Object Models on Mobile Robots

Xiang Li
Department of Computer Science
Texas Tech University, TX 79409
xiang.li@ttu.edu

Mohan Sridharan
Department of Computer Science
Texas Tech University, TX 79409
mohan.sridharan@ttu.edu

Shiqi Zhang
Department of Computer Science
Texas Tech University, TX 79409
s.zhang@ttu.edu

***Abstract*— Although mobile robots are increasingly being used in real-world applications, the ability to robustly sense and interact with the environment is still missing. A key requirement for the widespread deployment of mobile robots is the ability to operate autonomously by learning desired environmental models and revising the learned models in response to environmental changes. This paper presents an approach that enables a mobile robot to autonomously learn layered models for environmental objects using temporal, local and global visual cues. A temporal assessment of image gradient features is used to detect candidate objects, which are then modeled using color distribution statistics and a spatial representation of gradient features. The robot incrementally revises the learned models and uses them for object recognition and tracking based on a matching scheme comprising a spatial similarity measure and second order distribution statistics. All algorithms are implemented and tested on a wheeled robot platform in dynamic indoor environments.**

**Keywords: Visual learning; Recognition; Wheeled robots.**

## I. INTRODUCTION

Mobile robots are increasingly being used in a range of real-world applications such as health care and navigation [1], [2]. However, autonomous operation continues to be a major challenge to the widespread deployment of robots in the real world. The challenge is all the more formidable when visual input from color cameras is considered, due to the sensitivity to environmental factors and the computational complexity of visual input processing algorithms. As a result, despite being a rich source of information, vision remains under-utilized in many robot applications. For instance, many sophisticated algorithms have been developed for object recognition, a key component of a robot vision system. Unlike classical algorithms that require templates of target objects under different viewpoints, methods based on image gradient features are used extensively on mobile robots [3], [4]. However, gradient features are unsuitable for representing objects with texture-less surfaces (e.g., walls). In addition, many methods based on gradient features are computationally expensive and require an extensive training phase, while mobile robots operating in the real-world are faced with dynamic, unpredictable changes.

The above-mentioned challenges are offset by the fact that real-world domains are characterized by a significant amount of structure. Objects with unique characteristics (e.g., color and shape) may exist at specific locations, though these characteristics and locations may not be known in advance and may change over time. This paper describes an approach that enables a mobile robot to autonomously learn models for novel objects introduced in its environment. The approach draws its inspiration from nature: a chameleon that has camouflaged itself can be detected when it starts moving, and the human visual system is very sensitive to motion. The proposed approach therefore identifies candidate objects using temporal visual cues based on tracking gradient features over successive frames. Each candidate object is then characterized by image gradients and color distributions extracted from the corresponding image regions. The learned models are augmented with a spatial coherence vector and second-order statistics for robustness. The learned object models are then used in a probabilistic matching strategy to detect and track the corresponding objects in subsequent frames. The main advantage of the learning method is that it *bootstraps* off of the available information: the learned models are revised incrementally as the corresponding objects are recognized, leading to a more accurate recognition of these objects in subsequent frames. All algorithms are evaluated on mobile robots in dynamic indoor environments.

The remainder of this paper is organized as follows. Section II reviews some related work, while Section III describes the components of the proposed learning algorithm. Section IV describes the experimental platform and experimental results, followed by the conclusions in Section V.

## II. RELATED WORK

Vision research has provided sophisticated algorithms for object recognition. Schmid and Mohr [3] represented objects using gray-value invariants computed at interest points, and used a voting algorithm and semi-local constraints on test images. Lowe [5] used image gradient features, known as the scale invariant feature transform (SIFT), for reliable matching across different views of an object. Matas et al. [6] represented objects with an affine-invariant set of extremal regions, the maximally stable extremal regions (MSER). Liebe et al. [7] estimated object shape by probabilistically combining the information from different training samples using a probabilistic generalized Hough transform.

Object recognition using local gradient features or color is a key component of a robot vision system [1], [8], [9], [10]. Se et al. [4] enabled a mobile robot to use scale invariant visual landmarks to localize globally and build a 3D map of the

environment. Color-based visual features have also been used to enable a mobile robot to track moving obstacles [11]. Ess et al. [12] jointly estimate camera position and stereo depth while detecting objects and their trajectories based on visual information. Recent focus has been on the development of algorithms for unsupervised learning of object models. For instance, Roman et al. [13] proposed a hierarchical scheme that relies on the stability of a subset of features extracted from sensory inputs to perform an initial robust classification based on unsupervised methods. Parikh et al. [14] described an approach for unsupervised learning of hierarchical spatial structures from images, using a rule-based model and a graph-based representation for each rule. However, most of these methods are computationally expensive or require extensive prior knowledge for modeling the target objects. This paper focuses on autonomously learning object models using local, global and temporal visual cues.

## III. PROPOSED ALGORITHM

The proposed work is based on our observation that it is typically more important for a mobile robot to learn models for, and keep track of, the moving objects in its environment. Stationary objects can be considered as part of the background that the robot learns as it maps its surroundings [15]. The learning algorithm is hence triggered by object motion, and it extracts local and global visual cues from appropriate image regions to build the layered object model. The learned models are then used to recognize and track the desired objects in the subsequent images.

The proposed algorithm consists of four components: unsupervised detection and learning of object models (Section III-A); incorporation of spatial (feature) coherence vectors (Section III-B) and second-order (color) distribution statistics (Section III-C) for improved robustness; and the effective fusion of different visual cues for robust object recognition and tracking (Section III-D). Specific details are described in the sections below.

### A. Unsupervised Learning of Object Models

The first step in the proposed algorithm is the unsupervised detection of novel moving target objects. In computer vision literature, optical flow methods have been used to detect motion of pixels in an image sequence [16]. Here, the detection of motion is based on tracking image gradient features. Our prior research showed that a combination of an efficient feature detector (MSER [6]) and a reliable feature descriptor (SIFT [5]) results in reliable and efficient object recognition [10]. MSER-SIFT features were hence used to characterize objects and detect their motion.

The detection of object motion proceeds as follows. Consider, for instance, the images captured at time $t$ and $t+1$, i.e., $I_t$ and $I_{t+1}$. Consider the set of MSER-SIFT features extracted from these two images: $MS_t = \{ms_{t,1}, \ldots, ms_{t,N}\}$ and $MS_{t+1} = \{ms_{t+1,1}, \ldots, ms_{t+1,M}\}$, where each feature is a 128D vector. The features in these two sets are matched with each other using the nearest neighbor algorithm [17]. The matched features are then clustered based on their relative displacement between the two images. The underlying hypothesis is that unique features corresponding to a single object are likely to have similar relative motion between two consecutive images. Clusters with more than a threshold number of matched features are considered as candidate novel objects that are in motion. Convex boundaries are computed for each set of matched and clustered features. If a boundary includes many features that were not in the corresponding cluster, the cluster is removed from the list of candidate objects. In addition, the detection of candidate objects is made robust by performing the pair-wise matching over a set of three consecutive images.

The candidate objects are characterized using local gradient features and color distributions because of their complementary properties. Local gradients are robust to scale, orientation, viewpoint and illumination, but they neglect the global information in color images and are not appropriate for texture-less surfaces. Color features, on the other hand, provide a more global representation but do not have the robustness of local gradients. Each object model therefore consists of: (a) a set of MSER-SIFT features corresponding to the cluster of features matched across the set of images; and (b) a set of probability distributions (pdfs), where each pdf is a normalized histogram in the HSV color space. The color pdfs are generated using the image pixels within the convex boundary around the clustered gradient features corresponding to each object. The key fact is that the learning of object models is triggered by object motion and accomplished autonomously.

Though MSER-SIFT features are robust to many factors (e.g., scale and viewpoint), spurious matches can still occur because the features only consider a small (local) image region. For instance, the features from a wheel of a car may be similar to those from a different wheel or a different car. Similarly, color distributions of an image region need not be unique. If these learned object models are used for object recognition in subsequent images, there is a high likelihood that some of the matches would be incorrect. The learned object models are hence augmented using spatial arrangement of gradient features and second-order distribution statistics.

### B. Spatial Coherence Vector

Similar to the color coherence vector for color histograms [18], the spatial arrangement of local gradient features corresponding to a specific object is captured using a spatial coherence vector (SCV). The SCV is then used to compute a probabilistic spatial similarity measure (SSM) for object recognition in test images.

The motivation underlying the SCV computation is that though the individual features may not be unique, the spatial arrangement of features corresponding to a rigid object cannot be duplicated easily. The relative arrangement of the MSER-SIFT features corresponding to an object is hence used to increase robustness. The spatial coherence of each feature is defined as its position in the image relative to every other feature corresponding to this object. This relative

coherence is computed separately along the x and y axes. Let the object be characterized by $N$ MSER-SIFT features. The SCV for the $i^{th}$ feature is defined as:

$$SCV_{x,i} = \{d^x_{i,1}, d^x_{i,2}, \ldots, d^x_{i,N}\} \quad (1)$$
$$SCV_{y,i} = \{d^y_{i,1}, d^y_{i,2}, \ldots, d^y_{i,N}\}$$

where $d^x_{i,j}$ is the relative position of feature $i$ w.r.t feature $j$ along the x-axis; $d^y_{i,j}$ is the corresponding relative position along the y-axis. For instance:

$$d^x_{i,j} = \begin{cases} 1 & \text{if } x_i > x_j \\ 0 & \text{if } x_i = x_j \\ -1 & \text{if } x_i < x_j \end{cases} \quad (2)$$

where $x_i$ and $x_j$ are the x coordinate values of feature $i$ and $j$ respectively in the image plane.

The SCV computation is an additional layer in the representation described in Section III-A. Consider the situation described in Figure 1, where three SIFT features that share the same velocity $v$ are shown enclosed in a rectangular box.

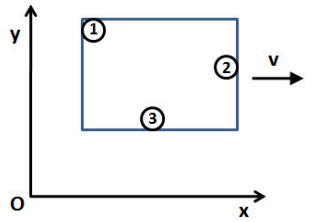


Fig. 1: SCV computation of MSER-SIFT features.

The SCVs of these three features along the x and y axes are shown in Table I and Table II.

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | – | −1 | −1 |
| 2 | 1 | – | 1 |
| 3 | 1 | −1 | – |

TABLE I: X-axis SCV

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | – | 1 | 1 |
| 2 | −1 | – | 1 |
| 3 | −1 | −1 | – |

TABLE II: Y-axis SCV

If the learned model for a novel object has $N$ MSER-SIFT features (each feature is a 128D vector), the model is augmented with a $2(N-1)$-dimensional vector for each feature that corresponds to the SCV along the x and y axes. As shown later, the augmented model results in robust object models and better object recognition.

### C. Color Distribution Statistics

For each candidate object, the robot extracts pixels within the corresponding image region to build normalized histograms, i.e., color space pdfs, in the HSV color space that is inherently robust to minor illumination changes. Pixel values in RGB are converted to HSV and normalized:

$$h = \frac{H/360}{H/360 + S + V} \quad s = \frac{S}{H/360 + S + V} \quad (3)$$
$$v = \frac{V}{H/360 + S + V}$$

where hue ($H$), saturation ($S$) and value ($V$) are the dimensions of the color space. After normalization, any two of the three dimensions are a sufficient statistic for pixel values. Each color pdf is hence modeled as a normalized histogram in the $(h, v)$ space, quantized into ten bins in each dimension.

As stated above, color distributions do not constitute a stable or unique representation for an object. Based on prior work on color learning on mobile robots [11], the robot computes the distance between every pair of pdfs and models the distribution of distances as a Gaussian. The distance computation is based on the Jensen-Shannon (JS) measure:

$$JS(a,b) = \frac{KL(a,m) + KL(b,m)}{2} \quad (4)$$
$$KL(a,b) = \sum_i \sum_j (a_{i,j} \cdot \ln \frac{a_{i,j}}{b_{i,j}}), \quad m = \frac{a+b}{2}$$

where $(a, b)$ are the two distributions (i.e., pdfs) and $m$ is a distribution obtained by averaging the two pdfs. The JS measure is based on the logarithm of pdfs, and it is robust to spurious peaks in the observed pdfs, e.g., due to large regions of a single color in the image under consideration.

This Gaussian distribution of distances is a second-order statistic that is included in the learned object model to represent the expected variance in the color distributions for the corresponding object. This statistic can also be used by the robot to detect and adapt to illumination changes, thereby making the system more robust [11].

### D. Information Fusion and Matching Strategy

As described above, the learned layered model for each object consists of color space pdfs, second-order distribution statistics, local gradient features and spatial coherence vectors corresponding to these gradient features. The learning algorithm is triggered by object motion, and the learned models are used for recognizing and tracking the corresponding objects in subsequent frames. In a new input image, the MSER-SIFT features are computed first and matched against the features in the learned models for different objects, using the nearest neighbor classification scheme. For the features in a learned object model, the nearest neighbors are found in the new input image. The SCV is computed for the matched features in the new image and compared against the SCV of the learned object model. A probabilistic spatial similarity measure (SSM) is introduced to compute the degree of similarity between the two SCVs:

$$SSM = \frac{N_{x,correct} + N_{y,correct}}{2 * (N-1)}, \quad SSM \in [0,1] \quad (5)$$

where $N_{x,correct}$ and $N_{y,correct}$ represent the number of values in the test image SCV that match the learned model's SCV along the x and y axes respectively, while $N$ is the number of values in the learned model's SCV. The SSM values range from $[0, 1]$, and $SSM = 1$ implies that the spatial arrangement of gradient features in the test image perfectly matches the arrangement of features of the corresponding learned object model. The SSM can hence be used as a probability measure of occurrence (in the test image) of the object under consideration.

A match measure is also computed using the color distribution features. For the set of matched features in the test image, the pixels within the convex boundary around

these matched features are used to compute a normalized color space histogram (i.e., a pdf). The mean (JS) distance is computed between this test image pdf and the set of pdfs corresponding to the learned object model under consideration. The deviation of this mean distance from the mean of the Gaussian distribution of the distances computed for this object model (i.e., the second-order statistic) provides a match probability based on the color distribution features.

**Algorithm 1** Object Model Learning and Recognition

**Require:** : Ability to learn object models based on gradient features and color distributions.

**Require:** Learned map of the surroundings for navigation.

1: Initialize: $numObjects = 0$ (no prior knowledge).
2: **while** true **do**
3:  **if** $modelLearn$ **then**
4:  Compute local gradient features for $I_t$ and $I_{t-1}$.
5:  **if** $DetectObject()$ **then**
6:  Compute SCV, color distributions and second-order statistics.
7:  **if** $numObjects > 0$ and $ObjMatch(objID)$ **then**
8:  $MergeLearnedModel(objID)$
9:  **else**
10:  $ComputeNewModel()$
11:  $numObjects = numObjects + 1$
12:  **end if**
13:  **end if**
14:  **else**
15:  Compute color and gradient features, SCV and second-order statistics for $I_t$.
16:  **if** $numObjects > 0$ **then**
17:  **if** $ObjMatch(objID)$ **then**
18:  State recognition of object $objID$.
19:  **end if**
20:  **end if**
21:  **end if**
22:  Track learned object models with Kalman filters [17].
23: **end while**

Assuming that there are $K$ learned object models, the SSM and JS distance-based matching schemes will each compute probabilities that the test image contains each of these objects. The net match probability is then given by the product of these two individual probabilities:

$$p_i = p_{ssm,i} \cdot p_{js,i}, \quad \forall i \in [0, K-1] \tag{6}$$
$$p_i = \frac{p_i}{\sum_{j=0}^{K-1} p_j}$$

where $p_{ssm,i}$ and $p_{js,i}$ are the probabilities, based on the gradient features and color features respectively, that the test image contains the $i^{th}$ object. A discrete distribution of match probabilities is hence computed over the range of learned object models, for the gradient features and color distributions. The net probability of match is obtained by multiplying the individual probabilities and normalizing the resulting distribution. This match probability can be used for two purposes: (1) to find the closest match and hence perform object recognition; and (2) to detect new objects when the probability of match is below a threshold, i.e., $p_i < probThresh$. Experimental results show that this strategy elegantly exploits the advantages of color features and gradient features to provide robust performance.

The overall algorithm is described in Algorithm 1. The robot begins with no prior knowledge of object models, though it has the ability to build models based on local gradients and color features given a specific image region. The robot has a learned map of the world (using range finders) for safe navigation, but has no initial learned model of the desired objects ($numObjects = 0$). If the robot is to learn object models ($modelLearn = true$ in line 3 of Algorithm 1), the robot considers the images captured at that time-step and the immediate preceding time-step ($I_t, I_{t-1}$). The MSER-SIFT features are extracted from these images and matched to arrive at the candidate object clusters (Section III-A). If a valid object is detected (line 5), the robot computes the spatial coherence vector, color distributions and second-order statistics to populate a model for that object. Next, if prior learned object models exist, the robot attempts to match the new model with an existing model (line 7 and Section III-D). If a match with a sufficiently high probability is found ($p_i > probThresh$ in Equation 6), the robot merges this new model with the existing learned model (line 8). However, if a match is not found, the robot stores a new model corresponding to this object and increments the count of learned object models (line 10-11).

If model learning is turned off, the robot can still perform object recognition if learned object models exist. The recognition can be done with a single image and even with a stationary object. The robot begins by computing the color and gradient features, along with the SCV and second-order distribution statistics for the current image ($I_t$)—line 15. The robot then attempts to match the computed model with one of the existing models. If a match with sufficiently high probability is found ($ObjMatch()$ in line 17), the recognized object is reported. Learning and recognition are separated in Algorithm 1 only for ease of explanation. As such, the robot can (and does) perform learning and recognition concurrently. In each frame, all detected object models are tracked using Kalman filters [10], [17].

## IV. EXPERIMENTAL SETUP AND RESULTS

This section describes the robot test platform and the results of evaluating the proposed algorithms on the robot.

### A. Experimental Platform

The ERA-MOBI robot platform (a.k.a "erratic") created by Videre Design [19] was used as the test platform. As shown in Figure 2, it is a compact robot base ($40cm \times 41cm \times 15cm$) equipped with a stereo camera, a monocular camera, laser range finders and a pan-tilt unit. The on-board processor is a 1.6GHz Core2 Duo with 1GB RAM. The experiments below use one of the cameras of the stereo unit, which provides images at a resolution of $640 \times 480$. The laser range finder on

Fig. 2: Robot test platform.

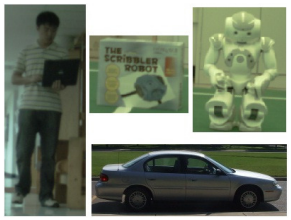

Fig. 3: Object categories used.

the robot is the Hokuyo UTM device with a maximum range of 30 meters, which can be used to map the environment. Though the robot has Wi-Fi capabilities to communicate with other robots or computers, all experiments were performed on-board the robot.

### B. Experimental Results

Four object categories were used in the experiments: humanoid robots, humans, boxes and cars—Figure 3 shows examples of each category. The category "car" was used for evaluation in outdoor environments, while the other three categories were used for indoor experiments. Different object models were learned for different objects within the same category. Figure 4 shows examples of objects within each category, which were used in the experiments. For instance, different boxes, humans, and the front and back of the humanoid robots result in different learned models. In addition, different colored patterns were taped to the waist of the humanoid robots to result in different object models. The objects were placed in environments with other background objects (see Figure 5) that made the learning of object models and the subsequent recognition quite challenging. During experimental trials, the mobile objects (i.e., human, car and humanoid robot) moved in random directions, while the boxes were moved on wheeled trolleys.

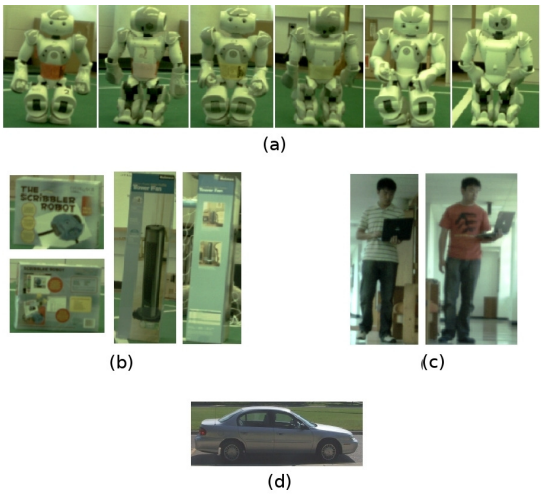


Fig. 4: Examples of objects that resulted in different learned models in each category: (a) Humanoid (b) Box (c) Human and (d) Car.

The experiments were conducted over a set of $\approx$ 200 images captured by the robot over a period of time. The robot was allowed to move during the experiments and used

Fig. 5: Experiments included indoor and outdoor environments.

image input to learn models for moving objects, and detect these objects in subsequent images irrespective of whether they were moving or stationary. As described in Algorithm 1, candidate objects are either merged with existing models or used to create new models. The robot also uses the learned models to detect the corresponding objects in test images.

Figure 6 shows a test image with a human walking down a corridor. At this point, the robot had already learned models for 13 different objects across the four categories. Figure 7 shows the result of using the SSM and JS distance-based matching scheme to compute the probability of occurrence of each of the learned objects in the test image. For ease of explanation, only the top five matches among the 13 available models are shown. The test images were different from those used to learn the object models—the images along the x-axis of Figure 7 are just snapshots of the object models. Figure 8 shows the merged probabilities for the top five matches, computed using Equation 6.

Fig. 6: Example test image.

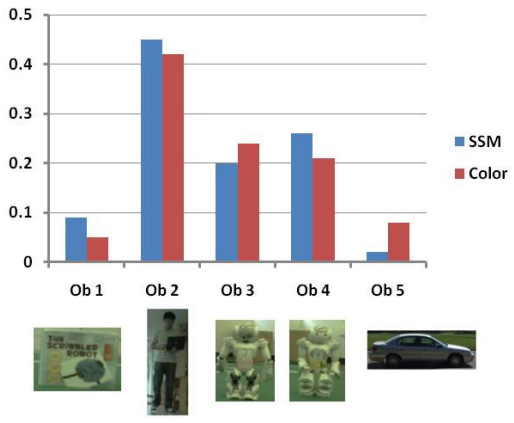


Fig. 7: Match probabilities based on SSM and color models.

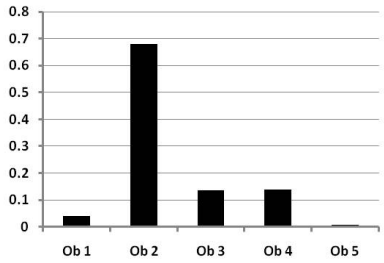


Fig. 8: The merged match probabilities for the test image.

The detection of novel objects is based on a threshold on the probability of match with the existing models. However, the method is not sensitive to the choice of threshold. For instance, when features extracted from images containing a "box" were compared with the corresponding learned model, the average match probability was high: $0.7668 \pm 0.1384$. However, when features from images without a box were compared with the box models, the average match probability was much lower: $0.3571 \pm 0.1227$. Similar results were obtained for other categories, and a threshold of $0.55$ was hence used in all the experiments.

| Category | Trials with object | Accuracy | Trials without object | Accuracy |
|---|---|---|---|---|
| Humanoid | 120 | 0.925 | 90 | 0.878 |
| Box | 110 | 0.936 | 100 | 0.928 |
| Human | 30 | 0.767 | 180 | 0.867 |
| Car | 30 | 0.733 | 180 | 0.833 |

TABLE III: The classification accuracy for each object category. Errors are due to within-class misclassifications.

Table III shows the classification accuracy averaged over the object models in each category. The classification is considered to be correct if the robot matches the object in the test image to the appropriate model within the corresponding category. In other words, if the robot matches an image of a human in *human-class1* to the learned model *human-class2*, it is considered to be incorrect. Most of the classification errors in Table III correspond to erroneous classifications within a category—an object is never assigned a label from a different category. Furthermore, images are analyzed at $3-7$ frames/second, and efficiency can be improved further by processing only the relevant regions of input images.

The true-positive accuracy was high for the "humanoid" and "box" categories, and the true-negative accuracy was high for all categories. The accuracy was lower in the "human" and "car" categories that involved many experiments in outdoor or cluttered environments. In these cases, the learned models had many non-unique features. For instance, the features in the shirt of one human were similar to the features extracted from the image of a different human. This problem predominantly occurred when the human or car was close to the camera—when observed at a reasonable distance from the camera, suitable models were learned for objects in all categories. One future direction of research is to compute motion cues based on other image features in order to learn more unique models for categories with a significant amount of within-category similarity.

## V. CONCLUSIONS AND FUTURE WORK

Autonomous operation is a key requirement for mobile robots in dynamic real-world domains. This paper described an algorithm that enables a mobile robot to autonomously detect candidate objects based on motion cues, and to learn layered object models based on gradient features and color distributions. Spatial coherence vectors and second-order statistics are incorporated for increased robustness. A probabilistic matching scheme based on the learned object models is used to recognize the corresponding objects in test images and track them in subsequent images.

Future work will incorporate other visual cues in the approach in order to achieve robust learning of unique object models. In addition, the experiments summarized above only had a couple of objects moving at a time. One direction of further research is to investigate the extension to several moving objects. Furthermore, the approach can be extended to a team of robots collaborating in dynamic environments.

## REFERENCES

[1] S. Thrun, "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[2] J. Hoey, P. Poupart, A. Bertoldi, T. Craig, C. Boutilier, and A. Mihailidis, "Automated Handwashing Assistance for Persons with Dementia using Video and a Partially Observable Markov Decision Process," *Computer Vision and Image Understanding*, vol. 114, no. 5, pp. 503–519, 2010.

[3] C. Schmid and R.Mohr, "Local grayvalue invariants for image retrieval," *PAMI*, vol. 19, no. 5, p. 530C535, 1997.

[4] S. Se, D. Lowe, and J. Little, "Global Localization using Distinctive Visual Features," in *International Conference on Intelligent Robots and Systems*, 2002.

[5] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[6] J. Matas, O. Chum, M.Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *British Machine Vision Conference*, 2002.

[7] A. L. B. Leibe and B. Schiele, "Robust object detection with interleaved categorization and segmentation," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 259–289, 2008.

[8] E. Bayro-Corrochano and C. Lopez-Franco, "Invariants and omnidirectional vision for robot object recognition," in *IROS*, 2005.

[9] P. J. S. Ekvall and D. Kragic, "Integrating active mobile robot object recognition and slam in natural environments," in *IROS*, 2006.

[10] M. Sridharan and X. Li, "Autonomous Information Fusion for Robust Obstacle Localization on a Humanoid Robot," in *International Conference on Humanoid Robots*, 2009.

[11] M. Sridharan and P. Stone, "Global action selection for illumination invariant color modeling," in *IROS*, 2007.

[12] K. S. A. Ess, B. Leibe and L. van Gool, "Moving obstacle detection in highly dynamic scenes," in *ICRA*, 2009.

[13] N. E. K. Roman, N. Juan and D. Bertrand, "Track-based Self-supervised Classification of Dynamic Obstacles," *Autonomous Robots*, vol. 29, no. 2, pp. 219–233, 2010.

[14] D. Parikh, C. L. Zitnick, and T. Chen, "Unsupervised learning of hierarchical spatial structures in images," in *CVPR*, 2009.

[15] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. USA: MIT Press, 2005.

[16] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Prentice Hall, 2002.

[17] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2008.

[18] Z. R. P. Greg and M. Justin, "Comparing Images Using Color Coherence Vectors," in *Fourth ACM International Conference on Multimedia*, 1997.

[19] V. Design, "Videre Design Robot and Sensors," 2010, http://www.videredesign.com/index.php?id=21.