# Robust Building Identification for Mobile Augmented Reality

Vijay Chandrasekhar, Chih-Wei Chen, Gabriel Takacs

***Abstract*—Mobile augmented reality applications have received considerable interest in recent years, as camera equipped mobile phones become ubiquitous. We have developed a "Point and Find" application on a cell phone, where a user can point his cell phone at a building on the Stanford campus, and get relevant information of the building on his phone. The problem of recognizing buildings under different lighting conditions, in the presence of occlusion and clutter, still remains a challenging problem. Nister's Scalable Vocabulary Tree (SVT) [1] approach has received considerable interest for large scale object recognition. The scheme uses heirarchical $k$-means to create a vocabulary of features or "visual words". We first show how we can use a SVT and an entropy-based ranking metric to achieve 100% recognition on the well known ZuBuD data set [2]. We present a SVM kernel-based extension to the SVT approach and show that it achieves a 100% recognition rate as well. We discuss the shortcomings of the ZuBuD data set, and present a more challenging Stanford-Nokia data set, with promising results.**

## I. Introduction

HIGH-END mobile phones have developed into capable computational devices equipped with high-quality color displays, high-resolution digital cameras, and real-time hardware-accelerated 3D graphics. They can exchange information over broadband data connections, and be aware of their locations using GPS. These devices enable many new types of services such as a car or pedestrian navigation aid, a tourist guide, or a tool for comparison shopping. For many of these services knowing the location is a critical clue, but that is not enough. For example, a tourist can be interested in finding information on several objects or stores visible from that location. Pointing with a camera-phone provides a natural way of indicating ones interest and browsing information available at a particular location. Once the system recognizes the target that the user is pointing at, it can augment the viewfinder with graphics and hyper-links that provide further information (such as the menu or customer ratings of a restaurant) or services (reserve a table and invite friends for a dinner). We call such a system a Mobile Augmented Reality (MAR) system.

### A. Stanford-Nokia MAR System

Figure 2 gives an overview of the system which is divided into two major components, a mobile device and a server. These components communicate over a wireless network. Once the user takes a picture, we extract SURF features [3] from the image and send them to a server. The server recognizes the building or the object in the picture by matching the image against a database of geo-tagged images. The server groups the geo-tagged images into location cells typically of


Fig. 1. A snapshot of the outdoors augnted reality system being used. The system augments the viewfinder with information about the objects it recognizes in the image taken with a phone camera.

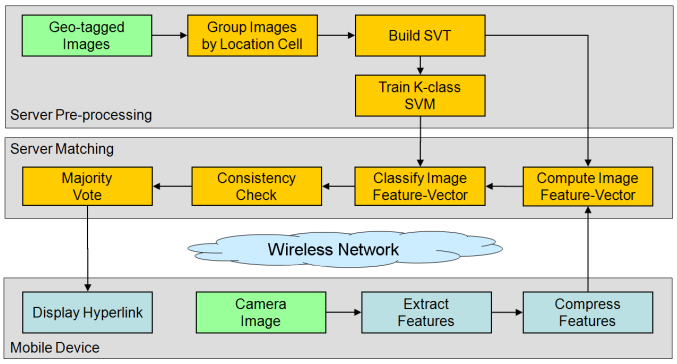


Fig. 2. System block diagram. The system is divided into two major components, a mobile device and a server, which communicate over a wireless network.

the size 1 km $\times$ 1 km. A location cell of this size would typically have 50-200 points of interest (buildings, restaurants, signs, etc). The recognition algorithms used are described in the following sections. The recognition algorithms return a ranked list of database images that are closest in content to an incoming query image. Affine geometric consistency checks are then carried out to remove spurious matches. In this paper, however, we will be focusing on the recognition algorithms that return a ranked list of images, prior to the geometric consistency check. Once the recognition is done, the server sends relevant information back to the phone. The phone finally overlays the information on the image, as shown

in Figure 1 above.

### B. Contributions and Paper Outline

This paper describes two methods for identifying the contents of the picture taken by a user with a camera phone. The algorithms described here allow recognition among a large number of classes or points of interest. A coarse estimate of the user's location via GPS or cell tower triangulation suffices to narrow down the search space (e.g., 1 km $\times$ 1 km). Even when a coarse estimate of the user's location is available, the search space may still be substantial and the user could be looking at one of many objects or buildings. We propose using a Scalable Vocabulary Tree (SVT) [1] in order to allow for image queries against a larger collection of images. The SVT allows us to define a bag of "visual words", analogous to a set of textual words. First, we use an entropy based distance metric proposed by [1] to rank the similarity of images in the database to a query image. We then propose a Support Vector Machine (SVM) extension of the SVT to learn the class membership by representing the different classes as bags of "visual words".

Section II discusses some of the prior work in this field. Sections III and IV describe the algorithms used for image matching on the server. Section V describes the two data sets used in the paper. Section VI evaluates the performance of the algorithms on the two data sets. We propose some enhancements VII to the schemes described here and conclude in VIII.

## II. Prior Work

Our work represents a special category of content-based image retrieval (CBIR) [4], [5], [6]. We want to recognize real-world images from a large set of categories (buildings, landmarks, logos) but we want to do it under a variety of viewing and lighting conditions. We want the algorithms to work in the presence of transient clutter in the foreground, and changes in appearance. Recent work in object-based image retrieval uses a vocabulary of "visual words" to search for similar images in very large image collections [7], [1], [8], [9]. The SVT approach has received considerable attention in the last year, due to its simplicity and its ability to categorize an object amongst a large number of classes.

## III. Scalable Vocabulary Tree

For the MAR application, we are interested in fast retrieval from a database of features containing noisy data. In this paper, we evaluate the bag of visual words approach for our application. There are two parts to the SVT; building the data structure for the images in the database, and then retrieving the closest match for an incoming query image.

### A. Hierarchical K-Means

An SVT is a data-structure that allows for efficient vector quantization and nearest-neighbor searching of feature vectors. The features in consideration are SURF features, which are stored as 64-dimensional floating point numbers. The vocabulary tree is built using hierarchical $k$-means clustering. A large set of representative descriptor vectors are used in the unsupervised training of the tree. Instead of defining the final number of clusters or quantization cells, $k$ defines the branch factor (number of children of each node) of the tree. First, an initial $k$-means process is run on the training data, defining $k$ cluster centers. The training data is then partitioned into $k$ groups, where each group consists of the descriptor vectors closest to a particular cluster center. The same process is then recursively applied. The path down the tree to any node can be encoded by a single integer, which is then used to represent that node or quantization cell. The hierarchical $k$-means, compared to the traditional $k$-means, is a scalable approach as it allows us to add features to the tree without having to rebuild the tree with the addition of new features.

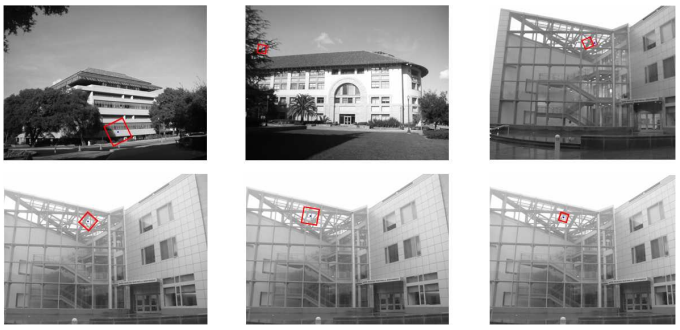

Fig. 3. Feature cluster. The features assigned to the same scalable vocabulary tree node are similiar.

On the server side, we group images into location cells of 1 km $\times$ 1 km. A SVT is built for all the features in each such location cell. We envision having 1000-10000 images in a location cell of this size. For this paper, we consider all the images in one such location with around a 1000 images in it. We extract features from all the images in the data set and insert them into the vocabulary tree. Each feature in the vocabulary tree is quantized to a leaf node of the tree. Each node of the tree stores a pointer to all the features that were quantized to the cell. The features, thus, point back to the images from which they were extracted. A typical cluster of features is shown in Figure 3.

### B. Scoring Metric

Once the quantization is defined, we wish to determine the relevance of a database image to the query image. The relevance score is based on how similar the paths down the vocabulary tree are for the descriptors from the database image and the query image. We take into considerations all the leaf nodes as well as interior nodes. Since interior nodes will be visited more frequently then leaf nodes, an entropy weighting $w_i$ should be assigned to each node $i$:

$$w_i = \ln \frac{N}{N_i}$$

where $N$ is the total number of features in the database, and $N_i$ is the number of features in the database that pass through node $i$. The entropy weighting in the metric gives less weight to the nodes that have more paths through them, and more weight to

the nodes visited less frequently. This is because nodes that are encounterd more frequently are less discriminative and carry less information about the image.

Having computed an entropy measure for every node in the SVT, we can then compute vectors that represent each database and query image. We define $q$ and $d$ to represent query and database vectors, respectively.

$$q_i = n_i w_i$$

$$d_i = m_i w_i$$

Here $n_i$ and $m_i$ are the number of descriptor vectors of the query and database image, respectively, with a path through node $i$. Thus, each image is represented by a vector with $T$ elements, where $T$ is the number of nodes in the vocabulary tree. A database image is then given a relevance score based on the normalized difference

$$s(q,d) = \| \frac{q}{\| q \|} - \frac{d}{\| d \|} \|$$

between the query and database vectors. Here, we consider an $L_1$-norm.

The two parameters that need to be fixed for the SVT are the branch factor and the depth. Based on the recommendations in [1], we fix the branch factor to 10 and the depth to 6. This generates 1M leaf nodes in the tree.

Once the relevance scores are computed, we have a ranked list of images. Additionaly, geometric consistency checks are carried out to figure out the correct match. The geometric consistency check is computed via a RANdom SAmple Consensus (RANSAC) algorithm that fits an affine model to the point matches in the query and database image. However, for this paper, we are interested only in the quality of the retrieval prior to the affine consistency check.

## IV. Support Vector Machine Extension

We treat this problem as a traditional classification problem and present a SVM extension to the SVT approach. The SVT approach allows us to define images as distributions (or histograms) of features. Let the cluster centers of the $T$ nodes of the vocabulary tree be $(p_1, p_2, ..., p_T)$.

We extract the features from each image, and represent the image as the distribution $(p_1, u_1), (p_2, u_2), (p_3, u_3), ..., (p_T, u_T)$ where the $u_i$'s are the number of features assigned to cluster $i$, divided by the total number of features. Thus $u_i$'s are the proportion of features in the image assigned to cluster $i$. Note that we use all the nodes of the SVT as "visual words", and not just the leaf nodes.

We compare two histograms by using the $\chi^2$ distance,

$$D(u,w) = \frac{1}{2}\Sigma_i \frac{(u_i - w_i)^2}{u_i + w_i}$$

We can train a SVM classifier with Gaussian kernels based on a $\chi^2$ distance between two image distributions. Such an approach has been used in the literature for texture and object recognition [10].

In a two-class case, the decision function for a test sample $x$ has the following form,

$$g(x) = \Sigma_i \alpha_i y_i K(x_i, x) - b$$

where $K(x_i, x)$ is the value of a kernel function for the training sample $x_i$ and the test sample $x$, $y_i$ is the class label of $x_i$ (+1 or -1), $a_i$ is the learned weight of the training sample $x_i$, and $b$ is a learned threshold parameter. The training samples with weight $a_i > 0$ are support vectors.

We define a Gaussian kernel function, as defined below.

$$K(S_i, S_j) = \exp\left(-\frac{1}{A} D(S_i, S_j)\right)$$

It is shown in the literature that the Gaussian kernel with a $\chi^2$ distance function is a Mercer kernel [10].

For a SVM, there are two parameters that need to be chosen carefully, the kernel parameter $A$ and the SVM regularization parameter $C$. The $C$ parameter is chosen using cross validation. We set $A$ to the mean value of the $\chi^2$ distances between all training images in the data set as suggested by [10]. The paper claims that choosing this value produces comparable results to the brute-force search with cross-validation. We choose this method since it is computationally cheaper.

As described in the system diagram, GPS information significantly narrows down the search space. However, the input image can belong to one of many point-of-interest classes. Standard SVMs are used in the two-class setting for binary detection. To extend the SVM to multiple classes, we use the one-against-one technique. The one-against-one technique trains a classifier for every pair of classes, and a majority vote is carried out to determine to which class a query image belongs. This approach also returns a ranked list of classes for an incoming query image. We use the one-against-one technique to avoid the problem of unbalanced data sets. Also, the one-against-one technique is said to perform as well as the one-against-other technique for multi-class SVMs [10].

## V. Data sets

### A. ZuBuD Dataset

The ZuBuD database [2] consists of 640×480 pixel, color images of 201 buildings in Zurich, each represented by 5 images acquired at random arbitrary view points. It also seems that the images were collected around the same time. The images in the training set and the test set are very similar to each other, as evident in the results shown in Figure 4. The authors of the ZuBuD database also created 115 query images (320×240) of a subset of the same buildings to test the performance of their recognition system. To our knowledge, six papers have reported results on the ZuBuD dataset with recognition rate ranging from 40.87% in [11] to 80% in [12] to 100% in [13]. Here recognition rate is defined as the percent of the time that the system's top choice is correct.

### B. Nokia-Stanford Dataset

The Nokia-Stanford outdoors image data set contains 1060 640x480 images taken with a camera-phone from Stanford Campus, Stanford Shopping Center, and Santa-Cruz Avenue.

TABLE I
TABLE: % OF TOP MATCHES CORRECT

| | SVT Entropy Scoring | SVM extension |
|---|---|---|
| ZuBuD | 100 | 100 |
| Stanford-Nokia | 72 | 52 |

The images have a wide coverage of 33 different points of interest. However, the dataset is intentionally non-uniformly sampled. The images are taken from difference seasons, under varying lighting conditions, from different angles, and with a lot of image clutter. With this dataset it is generally more difficult to achieve good matching performance. We use 33 query images for evaluating the performance of our algorithms.

## VI. EXPERIMENTAL RESULTS

To evaluate the performance of our algorithms, we consider two metrics. For the SVT ranking metric algorithm, we look at the percentage of correct matches for each one of the top 5 ranked images. Additionally, we also consider the percentage of images that have at least one correct match among the top 5 matches. The SVM algorithm returns a ranked list of classes aswell. We consider the percentage of correct class matches among the top 5 ranked classes.

For the SVM, the regularization parameter C was chosen by carrying out a brute force search. The value of C that gave the highest accuracy by using 5-fold cross-validation was chosen as seen in Figure 11. A modified version of the open source libSVM library [14] was used for the SVM algorithms.

In the ETH dataset, Figure 5 shows that all the images at the top of the ranked list are correct. However, the percentage of correct matches decreases as we go down the ranked list, shown in Figure 5. Since the top match is always correct, there is at least one correct match in the top 5 matches, shown in Figure 6. The SVM algorithm performs well on the ETH data set. As seen in Figure 7, for each query image, the class at the top ranked list is correct. We believe that this is because the training set and test images are very similar - this is most probably because the test data was collected at the same time as the training data. We observe that the SVM algorithm is faster than the SVT approach. This is because the SVM is $O$(Number of classes), while the SVT approach is $O$(Number of database images with at least one feature in common).

However, these approaches do not work as well on the Stanford dataset. We observe 70% of the top matches are correct in Figure 8. Also, just like in the case of the ETH data set, the percentage of correct matches decreases as we go down the ranked list, as shown in Figure 8. This is because the Stanford dataset is much more challenging and the images have a lot more occlusion and clutter. However, we do find atleast one correct match among the top 5 matches, shown in Figure 9. For the SVM, 52% of the top match classes are correct, and the correct matching class is found 84% of the time within the top 5 classes.

Partial occlusion of the building causes the distribution of features to change, thus affecting the entropy based scoring metric, as well as the SVM training.

The results have been tabulated in Tables I and II.


Fig. 4. Sample matching results for images from the Stanford and Nokia data set

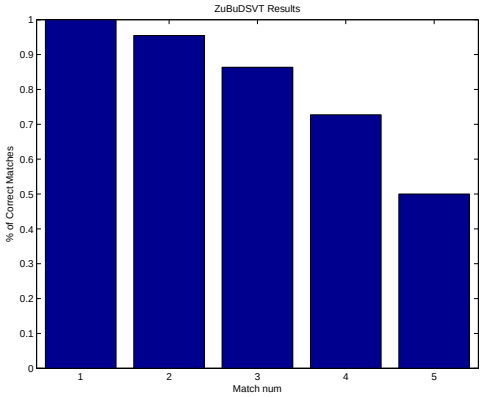


Fig. 5. ETH Data Set: Plot of % of correct matches vs. Rank. The results are for the SVT entropy based scoring approach. We observe that the first match is always correct, and the % of correct matches decreases as we walk down the ranked list.

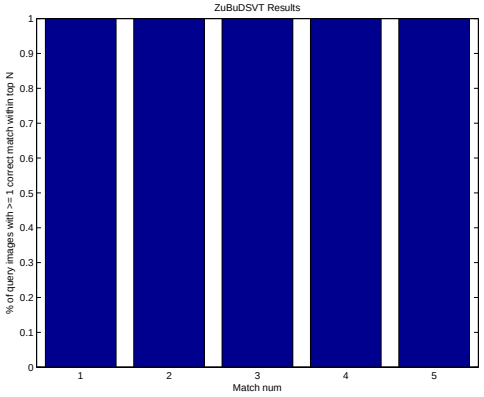


Fig. 6. ETH Data Set: Plot of % of query images with atleast one correct match within the top N ranks vs. Rank. The results are for the SVT entropy based scoring approach. Since, the first match is always corret, the plot stays constant at 100%

TABLE II
TABLE: % OF MATCHES CORRECT IN THE TOP 5

| | SVT Entropy Scoring | SVM extension |
|---|---|---|
| ZuBuD | 100 | 100 |
| Stanford-Nokia | 100 | 84 |

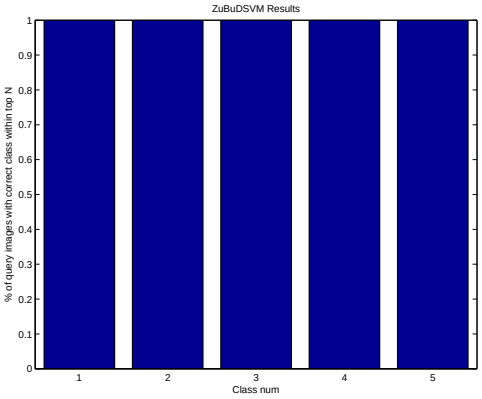


Fig. 7. ETH Data Set: Plot of % of query images with correct class within the top N ranks vs. Rank. The results are for a 200 class SVM with majority voting. We observe that the top ranking class is always correct.

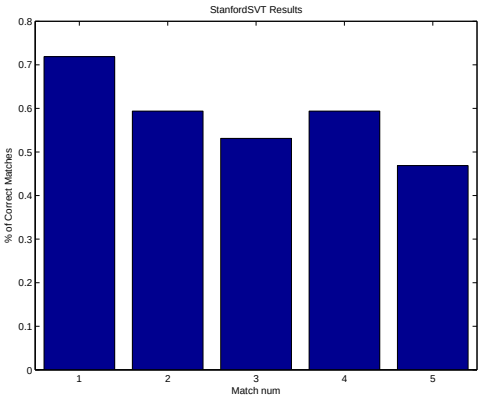


Fig. 8. Stanford Data Set: Plot of % of correct matches vs. Rank. The results are for the SVT entropy based scoring approach. We observe that the first match is correct 70% of the time, and the % of correct matches decreases as we walk down the ranked list.

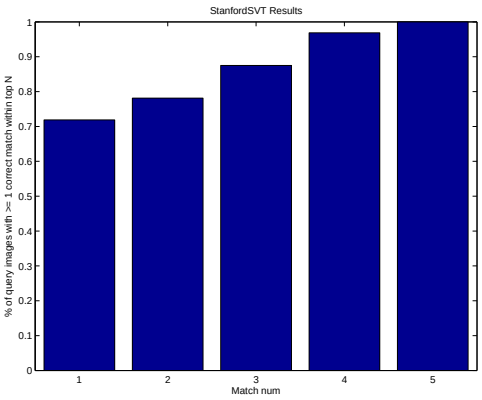


Fig. 9. Stanford Data Set: Plot of % of query images with atleast one correct match within the top N ranks vs. Rank. The results are for the SVT entropy based scoring approach. We observe that all query images have atleast one correct match within the top 5 images

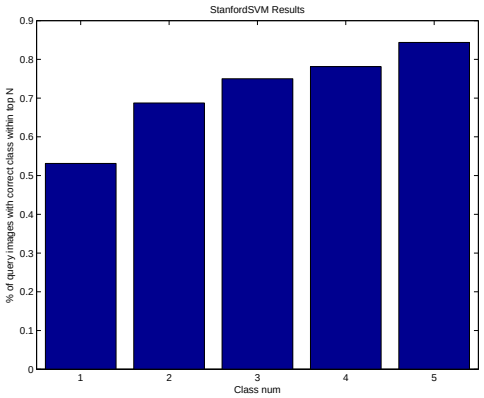


Fig. 10. Stanford Data Set: Plot of % of query images with correct class within the top N ranks vs. Rank. The results are for a 33 class SVM with majority voting. We observe that the top ranking class is correct 52% of the time. The correct match lies within the top 5 for 84% of the query images

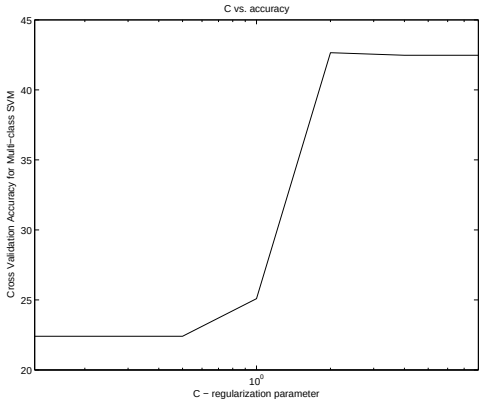


Fig. 11. Stanford Data Set: Plot of accuracy vs the SVM regularization parameter C. The value of C is varied from 0.0625 to 8. This is for Stanford data set for a 33 class SVM with majority voting. We observe the optimal value of C to be 2.

## VII. ENHANCEMENTS

We would like to improve the performance of the SVM and SVT algorithms on the Stanford-Nokia dataset. Currently, we have a relatively small number of images per class (on average 20-30) for the SVM training stage, and a total of only about a 1000 images. Our database is constantly growing as more people upload images. We are interested in performing experiments on data sets as large as 10000 images. The performance of the SVM might improve if more training data were available for each class [9].

Also, as user generated images tend to be noisy, we would like to learn the discriminative features for each point of interest. Similarly, we would like to use supervised and unsupervised machine learning techniques for learning foreground and background features in each image [15].

## VIII. CONCLUSION

We present algorithms for building recognition in a large database of images. We use hierarchical $k$-means to create a bag of "visual words". These visual words are used to train SVMs for different classes of buildings. For recognition, we consider two algorithms – an entropy based scoring metric based on the SVT, and a majority voting scheme based on a multi-class SVM. We test our algorithms on the well known

ETH building data set, and a more challenging Stanford-Nokia data set, collected from cell phone camera images in the last one year. We observe that both the SVT entropy based scoring, and the SVM algorithm work exceedingly well for the ETH data set (100%). The results are promising for the more challenging Stanford-Nokia data set. The SVT entropy based scoring algorithm finds the top match correct 70% of the time, and finds the correct match 100% of the time within the top 5 images. The SVM approach finds the correct class within the top 5 classes, 84% of the time.

## IX. Note

The following modifications were made after the poster session and led to a significant improvement in results for both approaches described in the paper.

1) Including interior nodes of SVT as "words" for the SVM approach.
2) Selection of regularization parameter via brute force search and n-fold cross validation.
3) Using libSVM package for SVM algorithms.

## References


[1] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2006, pp. 2161–2168.

[2] ETH-Zurich, *Zurich building image database*, available at http://www.vision.ee.ethz.ch/showroom/zubud/index.en.html.

[3] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in *ECCV (1)*, 2006, pp. 404–417.

[4] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-Based Image Retrieval at the End of the Early Years," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1349–1380, 2000.

[5] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture LIbraries," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 9, pp. 947–963, 2001.

[6] T. Yeh, K. Tollmar, and T. Darrell, "Searching the Web with Mobile Images for Location Recognition," in *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2. IEEE Computer Society, 2004, pp. 76–81.

[7] G. Fritz, C. Seifert, and L. Paletta, "A Mobile Vision System for Urban Detection with Informative Local Descriptors," in *ICVS '06: Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*, 2006, p. 30.

[8] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object Retrieval with Large Vocabularies and Fast Spatial Matching," in *Proc. of Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[9] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR07)*, Minneapolis, June 2007.

[10] J. Zhang, M. Marsza, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: A comprehensive study," *International Journal on Computer Vision*, 2006.

[11] V. T. T. V. G. L. Hao Shao; Svoboda, T.; Ferrari, "Fast indexing for image retrieval based on local appearance with re-ranking," *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 3, pp. III–737–40 vol.2, 14-17 Sept. 2003.

[12] J.-H. Lim, J.-P. Chevallet, and S. Gao, "Scene identification using discriminative patterns," in *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 642–645.

[13] J. Matas and S. Obdrzalek, "Object recognition methods based on transformation covariant features," in *EUSIPCO 2004*, September 2004. [Online]. Available: http://cmp.felk.cvut.cz/ matas/papers/matas-eusipco04.pdf

[14] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[15] D. Liu and T. Chen, "DISCOV: A Framework for Discovering Objects in Video," in *IEEE Transactions on Multimedia*, 2008.