RECOGNIZING REAL WORLD OBJECTS USING MULTIPLE VIEWS

Costas Cotsaces and Nikos Nikolaidis

Aristotle University of Thessaloniki, Department of Informatics
University Campus, Box 451, GR-54124, Thessaloniki, Greece
and
Informatics and Telematics Institute, CERTH, Thessaloniki, Greece

email: {*cotsaces,nikolaid*}*@aiia.csd.auth.gr*

ABSTRACT

Object recognition, i.e. the classification of objects into predefined categories is an important tool in many computer vision systems. Unlike other types of recognition, it must be quite generic in order to be able to handle the great variety of objects that can exist. An aid to the solution of this difficult problem can be the use of information from different camera views. Here, we have extended a robust object recognition method in order to be able to function with information from more than one camera, and from arbitrary viewpoints. This method uses local feature points to construct visual vocabularies which then form an input to Support Vector Machines. We have found the multi-camera variant to produce superior results to the single-camera one.

1. INTRODUCTION

Object recognition is quite unlike most other tasks in computer vision, for example face recognition, person detection, emotion recognition etc. This is because the word "object" may encompass a vast variety of different entities, both ontologically and visually. Additionally, many objects have a high degree of concavity, for example furniture or tableware. A significant minority of objects are also topologically complex (i.e. have holes in them). Moreover, in the case of objects, the border between recognition ("which one") and categorization ("what type of") is very blurry. This is because, unlike humans, objects do not have a distinct identity, and many objects can have multiple nearly identical copies (e.g cars of the same model). For the above reasons, object recognition and categorization algorithms need to follow one of two paths. One is to focus on one class of object e.g. cars, desktop items etc, and develop an algorithm that exploits the specific characteristics of this object class. The other is to be generic enough to cover many different types of objects.

Another characteristic of object recognition is that, with the exception of a few classes of radially symmetric objects, most objects exhibit a high visual variety from different views, and that in general the discriminant power of different views is generally similar. This is in contrast to areas such as person detection and recognition. For example, a car's side, front and rear view are about equally useful when trying to recognize different cars, whereas the rear view of a person is useless when trying to recognize him.

For the above reason, the fusion of information from multiple cameras is of great help in object recognition. Thus, we have attempted to design a system that performs object recognition (or categorization) using information from (initially) two cameras. Instead of designing a system from scratch, we chose to extend a successful generic single-camera object recognition and categorization system [3] which has been recently shown [7] to have a performance close to the state of the art.

Surprisingly, there has been little published work on the merits of multi-camera approaches to object recognition. The work of Christoudias et al. [2] is the most similar to ours, since they also use local descriptors, but they only use a simple nearest neighbor recognition scheme, and focus on feature selection. Campbell et al [1] can also be considered to perform multi-camera object recognition, using an object's color and edge information. Both works however do much of their testing using synthetic images.

In the following, in Section 2 we describe the single-camera method that has been extended. Then in Section 3 we give the multi-camera framework that was applied. The experimental results are given in Section 4, while conclusions and a few directions for future work are given in Section 5.

2. SINGLE CAMERA FRAMEWORK

The single camera method used was founded on the current trends in computer vision, specifically the use of local feature points, and decision using Support Vector Machines (SVMs). Since we wanted a method that is generic and not limited to a specific domain, we chose to base single-camera recognition on the work of Csurka et al. [3], which is sufficiently generic. Additionally, it has been shown by Zhang et al. [7] to have a superior performance in many tasks, especially object class recognition, showcasing its robustness as a base for object recognition.

The fundamental steps for the training phase of the single-camera method are the following:

1. Extraction of local feature points from all (labelled) images of a training set
2. Clustering of local feature descriptors into a number of classes
3. Computation of summary descriptor (feature vector) for each image in the training set
4. Training of a classifier using the feature vectors of all images

Correspondingly, the steps in the testing phase of the algorithm are the following:

1. Extraction of local features from the image that is being tested

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n^o 211471 (i3DPost).

2. Computation of the summary descriptor of the image
3. Classification of the image based on its descriptor, through the use of the previously trained classifier

In the following the implementations of each of the above steps will be addressed, and justifications for these implementations will be given.

The use of local feature points (also known as *local descriptors* or *keypoints*) as a basis for the description of an image is a common solution to the problems of occlusion, clutter, and changes in illumination, scale and rotation. Local features consist of two generally independent components, a feature point detector, and a feature point descriptor. In the present case, the feature point detector that was selected was the Harris affine detector [5]. Harris affine feature points are detected by an iterative process, by repeatedly computing local maxima of the Harris function of the image and estimating elliptical neighborhoods thereabout. The advantage of this detector is that is especially robust to transformations. The feature descriptor, respectively, is the classic SIFT descriptor [4]. This is a set of Gaussian derivatives computed at 8 orientation planes over a 4×4 grid of spatial locations, giving a 128-dimensional vector. Its discriminance and invariance has been well proven.

The clustering of local feature descriptors is done in order to abstract the distribution of the feature points. At this stage, it is not necessary to derive semantically significant clusters; all that is necessary is to give some granularity to the distribution of the descriptors. Thus, the simplest possible classification algorithm was selected, namely k-means. The number of classes k is decided experimentally, and generally ranges around 1000. The feature that is used for the assignment of a specific feature point to a cluster is the 128-dimensional feature point SIFT descriptor. At the end of the clustering procedure, only the cluster centers are retained.

The information that has been consolidated through the clustering needs to then be explicitly formulated into a feature vector for image that will be used for training the final classifier. For this, again the simplest possible method was chosen, i.e. a histogram representing the number of feature points that were assigned to each class center is constructed. Simplicity was chosen largely for reasons of robustness.

Finally, having characterized each image with a feature vector (the aforementioned histogram), all that remains is to train a classifier and use that classifier for a task of our choice. Depending on the separation of the training example images into classes, this could be object recognition, object class recognition (a.k.a. object categorization), object verification etc. In the present case, as stated above, we have chosen to implement object recognition. The classifier that was selected was the classic Support Vector Machine (SVM). Different types were tried, but once again the simplest linear variety was found to be most effective. Since SVMs are intrinsically a two-class classifier, we use the classic multi-class extension whereby a classifier is trained for each pair of classes and the final recognition of an image is done by a voting procedure, with classifier contributing a vote to the class it selects.

For recognizing an object in an incoming image (testing stage), a similar procedure is followed: Harris-affine featur points are detected, and their SIFT descriptors are extracted. These descriptors are then assigned into the previously computed cluster centers, and the number of feature points assigned to each center forms a histogram, which is then passed to the previously trained Support Vector Machine which makes the final decision about which object is depicted in the image.

3. MULTICAMERA FRAMEWORK

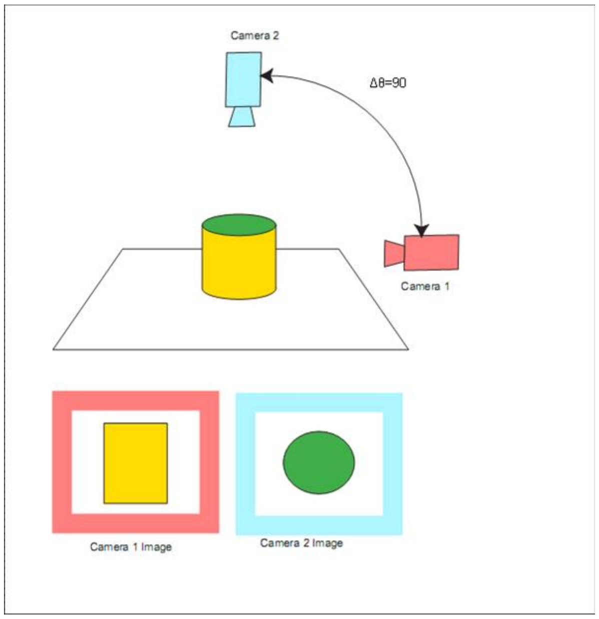

Figure 1: Schematic of the two-camera configuration used.

We base our approach to multi-camera object recognition on the assumption that the relative spatial configuration of the cameras is known a priori. Such an assumption is not unreasonable either in an experimental or in a production context, since in the former it is part of the experimental setup and in the latter it can be inferred by a number of camera calibration techniques. Of course, the above assumption limits us to *static* cameras, but even so the problem addressed is a significant one. Since the problem is object recognition, we are concerned only with the positions of the cameras with respect to the object, i.e. their relative position in a coordinate system rigidly attached to the observed object.

Additionally, we will limit ourselves to the two-camera problem. The reasons for this are multiple:

- It is easier to extract quantitative results and compare them with the single-camera case
- It is easy to formulate a multi-camera problem as a series of two-camera problems using a voting procedure similar to how we merge single-camera recognizers in this work.
- When multiple cameras exist it is reasonable to expect that two of them (the ones placed so that their axes are perpendicular to each other) will contain the most discriminative information.

Let us then assume cameras C_1 and C_2 as in Figure 1. Assuming a spherical coordinate system centered on the center of the object, the positions of the two cameras are $\{\rho_1, \phi_1, \theta_1\}$ and $\{\rho_1, \phi_1, \theta_1\}$. Since the key point bag recognition method is largely scale invariant, the only relevant parameters in this case are the angle differences, normalized to lie between 0 and 2π

$$\Delta\phi = \begin{cases} \phi_1 - \phi_2, & \text{if } \phi_1 \geq \phi_2 \\ \phi_1 - \phi_2 + 2\pi, & \text{if } \phi_1 < \phi_2 \end{cases} \tag{1}$$

and

$$\Delta\theta = \begin{cases} \theta_1 - \theta_2, & \text{if } \theta_1 \geq \theta_2 \\ \theta_1 - \theta_2 + 2\pi, & \text{if } \theta_1 < \theta_2 \end{cases} \tag{2}$$

Thus, in the general case, a different classifier would need to be trained for each combination of $\Delta\phi$ and $\Delta\theta$, i.e. for each spatial configuration of the two cameras.

The classifier that was used is a modification of the basic keypoint bag system described in Section 2. In essence, instead of a single keypoint histogram corresponding to the one camera, two keypoint histograms H_1 and H_2 are created, from the images corresponding to the two cameras. Each histogram is, as before, computed by classifying the SIFT descriptors into pre-computed bins. H_1 and H_2 are then concatenated into a combined histogram H, which is then processed as usual by the SVM stage.

The main difference is in the training stage. Firstly, in the training data, each training image X is labeled not only with the identity of the object in question, but also with the angles ϕ_x and θ_x from which it was imaged. An arbitrary "frontal" pose is chosen as a zero axis. Then, local SIFT features are extracted and the k-means centers are established as described in Section 2, without taking camera orientation into account. Likewise, the histograms for each image are computed irrespective of camera orientation, using the class centers generated by the k-means. But in order to compute feature vectors that will form the training data of the SVM, for each image X in the training database, other images Y belonging to the same class are sought such that

$$\phi_y - \phi_x = \Delta\phi + 2n\pi, n \in \{0,1\} \tag{3}$$

and

$$\theta_y - \theta_x = \Delta\theta + 2n\pi, n \in \{0,1\} \tag{4}$$

. If many such images are found, a random one is selected, and if none are found, the training image is discarded. Then the histograms of the two images H_x and H_y are concatenated into histogram H which is used to train the SVM classifier.

4. EXPERIMENTAL RESULTS

4.1 Databases used

When deciding which database to use, we initially considered using a pre-existing object database. However, databases that provide multiple labeled views of objects generally have some disadvantages. Specifically, they contain objects that are topologically simple, convex, and compact. This is not representative of objects in the real world (such as furniture), and does not demonstrate the strengths of our method in handling difficult objects. Additionally, most such databases are not recorded in realistic conditions with respect to lighting, shadows and background clutter. There are databases that provide such realistic conditions, such as PASCAL, but unfortunately they are not multi-view.

Thus, in order to provide a challenging corpus for our experiments we created our own database. We selected to focus on furniture, and particularly on chairs. Chairs are a recurrent object in human environments, and present particular difficulties since they are topologically very diverse, highly concave, and have large gaps in their silhouette.

For our training set, 8 distinct chairs were selected as shown in Figure 2. For each chair, 16 different camera orientations were defined, by sampling ϕ by 45° increments and selecting two different values of θ, 0° and 45°. In all cases, r was constant at $1.5m$ Within each orientation, 6 different photographs were taken, by randomly varying ϕ and θ within $\pm 10°$ and r by $\pm 15cm$. We thus gathered 96 images per object. For the sake of realism, the images were not taken against a completely blank background. However, care was taken to avoid the presence of other objects or significant background clutter. In general, the objects in question took up approximately 50% of the area of each image (including gaps in the objects).

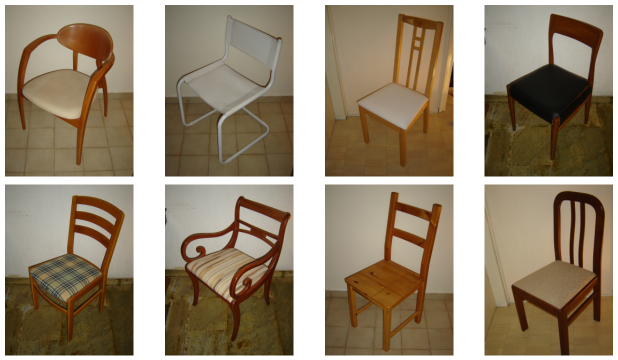

Figure 2: Example of the images captured for the chairs database.

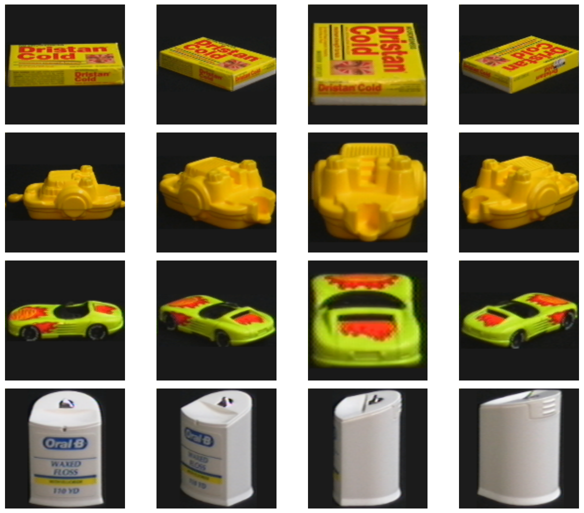

Figure 3: Example of some images from the COIL database.

However, in order to provide results comparable to other algorithms, we also tested our method on the standard COIL-100 database [6], which consists of simple mostly convex objects taken upon a black background. It comprises 100 objects taken from 72 different angles in 5° degree increments, but having no variation in the θ direction, examples of which are shown in Figure 3.

4.2 Experiments

The goal of our experimental setup was the comparison of the performance of the single camera baseline method with the multi-camera one. A straight comparison of the accuracy rates would be unfairly favorable to the multi-camera method, since it has access to more information than the single-camera one. A fair comparison would be between the result of the multi-camera recognizer and the result of the merging of two single-camera classifiers, each operating on one of the two images that form the input to the multi-camera recognizer. This is the approach that we have followed here. Specifically, when the two single camera results are different, the merging is achieved by selecting the one with the greatest SVM margin.

In our own database the goal was to recognize which chair is shown in each picture. We trained both methods using a random subset, containing $2/3$ of the images. In the case of the single-camera method, all images in this subset were used whereas for the multi-camera one, the images were matched into pairs. Then the other $1/3$ of the images were used for testing. This is repeated three times.

The result of the combination of the two single-camera classifiers was an accuracy of 96%, while the two-camera classifier had an accuracy of 100%, having made no mistakes. There was no detectable difference in execution time — it remained at about 2sec on a single core of an Intel Core 2 computer running at 2.33 GHz.

In the case of COIL, we also performed experiments using 2/3 cross-validation as in the previous section. The two-camera version achieved a recognition rate of 99,4% while the merging of single-camera classifiers resulted in a recognition rate of 97,2%. The above results show that the proposed two-camera extension improves the single camera baseline algorithm, achieving almost perfect recognition at least for the databases that were used.

5. CONCLUSIONS

We have extended an effective single-camera object recognition method to the two-camera problem, by explicitly including the relative position of the two cameras into the training of our classifier. Our experimental results show a clear improvement on a fairly difficult dataset, as well as a standard one. It should be noted here that the method can be readily extended to an arbitrary number of cameras. Further work may include the adaptation of this framework to work not only for recognition of objects but also for detection in cluttered scenes.

REFERENCES

[1] N. D. F. Campbell, G. Vogiatzis, C. Hernández, and R. Cipolla. Automatic 3d object segmentation in multiple views using volumetric graph-cuts. *Image Vision Comput.*, 28(1):14–25, 2010.

[2] C. M. Christoudias, R. Urtasun, and T. Darrell. Unsupervised feature selection via distributed coding for multi-view object recognition. In *Interntational Conference on Computer Vision and Pattern Recognition (CVPR 08)*, 2008.

[3] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[5] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. *Computer Vision ECCV 2002*, pages 128–142, 2002.

[6] S. Nene, S. K. Nayar, and H. Murase. Columbia object image library (coil-100). Technical report, 1996.

[7] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. Comput. Vision*, 73:213–238, June 2007.