MVA2007 IAPR Conference on Machine Vision Applications, May 16-18, 2007, Tokyo, JAPAN

Patch Based Localization of Visual Object Class Instances

Alexandra Teynor and Hans Burkhardt
Universität Freiburg
Chair of Pattern Recognition and Image Processing
79110 Freiburg, Germany
{teynor,burkhardt}@informatik.uni-freiburg.de

Abstract

Huge image databases require the automatic analysis of image content in order to retrieve information. Especially the detection and localization of visual object class members is an important issue. In this work, we deal with the localization of visual object class members in a patch based object recognition framework. In particular, we show how not only the location and scale of an object can be determined, but also the orientation, a parameter typically neglected in current localization systems. Our method uses features computed at Difference of Gaussian interest points and remembers the orientation of the local patches relative to the reference object. Using a general Hough transform like voting scheme, the position and orientation of query objects can be retrieved. Tests on two different leaf databases show the capabilities of the approach.

1 Introduction

Today, an enormous mass of digital image data is stored in big archives, e.g. at publishing companies, news agencies and also on our home desktop computers. Computer assisted retrieval systems are necessary in order to find data again. Content based image retrieval (CBIR) methods have shown to be successful in searching for images based on their pixel content only. Researchers now focus on the recognition of instances of visual object classes like cars, cows or airplanes. To achieve simple scene understanding, it is beneficial to also localize objects in an image. Methods using the local appearance of image patches have shown to be very successful for classification, so we investigate the question of localization also in this context. In addition to the position and the scale of the object in question, we also consider the orientation, which is neglected by most other localization approaches. This information can help future image retrieval systems to evaluate the relative position and orientation of objects in an image better. Of course, taking into account the orientation of objects is not necessary for all types of objects, since cars e.g. are likely to be found with the wheels on the ground. However, the orientation of other objects, like e.g. leaves, is not fixed.

The outline of this paper is as follows: first, we describe work related to our method in section 2, then explain the feature extraction and localization procedure in section 3. Tests and results can be found in sections 4 and 5. In section 6 we set out our conclusions.

2 Related Work

Using local information at specific points in an image has shown to be successful for object class recognition, since it can deal with object shape variability and partial occlusions. Recently, a variety of methods has been proposed. They can mainly be divided into two categories: on the one hand approaches using only feature cluster frequencies without location information like Csurka et al. [2] or Deselaers et al. [3], on the other hand methods also incorporating the spatial information of the patches.

Examples of the latter are Fergus et al. [6] and Fei-Fei et al. [5]. They use a so called "constellation model", i.e. specific local image features in a probabilistic spatial arrangement. Leibe et al. [7] introduced a joint classification and segmentation method for visual object classes. Their generalized Hough transform [1] like voting approach is related to our method, however, the rotation of the objects is not considered by them. A method that directly influenced our approach was presented by D. Lowe [8] for the detection of identical objects: the Scale Invariant Feature Transform (SIFT), however, we deal with object classes.

Most object recognition systems determine at most the position of visual object class members, not their orientation, so our algorithm offers extended functionality. Only recently, Mikolajczyk et al. [9] introduced a framework that is also capable of identifying the orientation of objects.

3 Method

Our method consists of two main parts: the creation of a model database using training images and the localization procedure itself. It is summarized in table 1, a more precise description can be found in the next section.

3.1 Feature extraction and training

In order to determine the local appearance of an object, we extract Difference of Gaussian (DoG) interest points in the images and compute SIFT features at these locations. We use the program provided by D. Lowe for this task. For each key-point, we get a location as well as size and orientation information. The orientation is the dominant gradient direction of the patch. We relax the discriminativity of the 128 dimensional SIFT descriptors while still coding the main characteristics of the patch: we perform a PCA on the feature vectors and only use a reduced set of coeffi-

Table 1: Summary of the algorithm

Feature extraction and training: For all training images:

1. Extract DoG interest points from the image. Discard interest points not related to the object. This can be achieved e.g. by using a segmentation mask of the object or other automatic procedures [4].
2. Determine SIFT features for each interest point coding the local appearance. Reduce the dimensionality by using a PCA (principal component analysis).
3. For each interest point, calculate the position of the object reference point relative to the interest point location, orientation and scale. Normalize the object scale with the interest point scale. Store this as geometry information.
4. Store the appearance features in a kd-tree or best-bin-first tree, retain a link to the specific geometry vector. This forms the model database.

Localization procedure: For a testing image:

1. Extract DoG interest points and calculate appearance features as in the training images.
2. Use the appearance vectors of each interest point for a nearest neighbor search in the model database.
3. Use the scale, position and orientation of the interest points together with the geometry vector associated to the respective nearest neighbors to determine hypothetic object positions, locations and scales. Store the votes in a 4D-Hough array.
4. Determine the maximum in the Hough space to get a possible object position, location and scale.
5. Perform a stability check on the solution found, otherwise search for the next maximum in the Hough array.

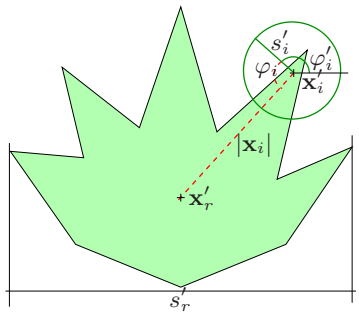

Figure 1: Geometry information extraction

cients. Using $2/3$ of the dimensions turned out to be sufficient in experiments.

Together with the appearance information, for every interest point p_i in a training image, a vector $\mathbf{v}_i$ coding the geometry gets extracted. It contains the distance of the point to a reference point, the direction to the reference point and the size of the object, relative to the interest point scale and orientation.

$$\mathbf{v}_i = (d_i, \varphi_i, s_i)$$

$$\begin{aligned} \mathbf{x}_i &= \mathbf{x}'_r - \mathbf{x}'_i \\ \varphi_i &= \arg(x_i + y_i \cdot \sqrt{-1}) - \varphi'_i \\ d_i &= |\mathbf{x}_i| / s'_i \\ s_i &= s'_r / s'_i \end{aligned}$$

with $\mathbf{x}'_r$ and s'_r being the object reference point and scale, $\mathbf{x}'_i$, s'_i, φ'_i the location, scale, and orientation of the interest point p_i, and $\mathbf{x}_i = (x_i, y_i)$. The superscript $'$ is attached to the directly measured values. These parameters are visualized in figure 1. In our experiments, we determine the center of gravity as the object reference point. Other points, e.g. the center of a bounding box could have been chosen as an alternative. For training, images containing the objects in a reference orientation ($0°$) and scale are used.

The interest points are not restricted to the shape (outline) of the object, but may lie on any characteristic part. The appearance vectors of all training images are stored in a kd-tree to allow for a fast nearest neighbor search.

3.2 Localization procedure

For the localization of rotated objects in a query image, interest points and features get extracted in the same way as for the training images. We use the appearance vector for a point p_j to retrieve the k nearest neighbors (with $k = 3$ in our case) in the reference database. The geometry vector of each nearest neighbor (now referenced with $\mathbf{v}_{j,n}$) is used to establish a hypothesis of a possible object location, orientation and scale, relative to the query interest point p_j:

$$\begin{aligned} \tilde{d}_{j,n} &= d_{j,n} \cdot s'_j \\ \tilde{x}_{j,n} &= \tilde{d}_{j,n} \cdot \cos(\varphi'_j) \\ \tilde{y}_{j,n} &= \tilde{d}_{j,n} \cdot \sin(\varphi'_j) \\ \tilde{\varphi}_{j,n} &= \varphi_{j,n} + \varphi'_j \\ \tilde{s}_{j,n} &= s_{j,n} \cdot s'_j \end{aligned}$$

where the ˜denotes hypothetic parameters and $n \in \{1, 2, \ldots, k\}$ the index of the current nearest neighbor.

All parameter sets vote for an entry in a 4D accumulator array as in a generalized Hough transform [1]. We employ a fuzzy voting approach disseminating the vote not only to the exact bin, but also to the neighbors to cope with small object deformations. Matched patches with a distance above a threshold get discarded. In order to localize an object, we search for maxima in the Hough space. The voting process is visualized in figure 2: in the first image, all interest points with their scale and orientation are shown. The features extracted at these points are used for a k-nearest neighbor search. In the second image, the resulting votes for possible object centers are displayed (all scales and rotations simultaneously). The hypothetic center points are connected with the respective interest points by a red line. The last image shows the localization of the object bounding box, together with the interest points supporting this hypothesis.

3.3 Eliminating unstable votes

Sometimes many votes for a specific parameter combination come from a single direction. These points are very unstable and might result from clutter in the scene. We favor parameter sets that get votes from different directions. To verify this, we also record the directions the votes came from. We quantize the angles into n sectors and require the votes to come from at least two non neighboring sectors, ensuring a minimum angle between voting interest points of

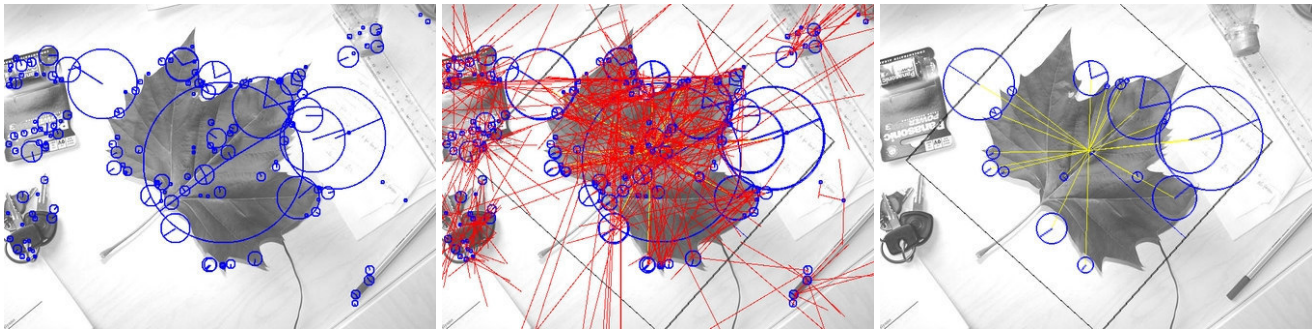

Figure 2: Localization process

at least $\frac{2\pi}{n}$. In our case, we use 8 sectors resulting in a required angle of minimum $45°$.

4 Experiments

For our experiments, we decided to use leaves as objects since they are likely to be found in any rotational position in an image and are naturally to be found in many variations. For the first experiments, we use the Caltech leaf database[1]. There are 6 different leaves from 3 tree types, each leaf photographed 10 times in a cluttered office background, resulting in a database of 180 images.

We rotated the test images randomly and recorded the angles. Since DoG interest points found along edges are removed by the SIFT detector, the newly introduced edges in the resulting images do not affect the localization procedure. To also have a database with naturally rotated images for our experiments, we created a database by ourselves, where the leaves were photographed in fixed orientations in front of similar office backgrounds. We photographed 10 different leaves of a tree in 8 orientations and in 7 different scenes, resulting in 560 images. The images are of size 640x480. Only grayscale information was used.

Training was performed on images with the objects in a reference orientation and size. The training leaves were segmented in order to calculate the center of gravity as the object reference point and to avoid background patches in the database. We performed cross-validation tests with a leave-one-object-out approach, i.e. the same leaf instance was never used for training and testing, despite being photographed in front of different backgrounds.

For all experiments, the Hough space was quantized to 30 bins for the x direction, the y direction and the angle, as well as 10 bins for the scale. For the Caltech leaf database, the experiments were performed on the 3 leaf classes separately as well as on the entire dataset. This shows that the approach is capable of localizing objects despite having more than one object class in the reference database.

We tested the best descriptor dimensionality, the accuracy of the estimated object reference point, the orientation of the object as well as the total localization accuracy (position and orientation). Scale changes were not tested explicitly, since all leaves in all databases are about the same size.

1 http://www.robots.ox.ac.uk/~vgg/data3.html

Table 2: Correct position in % and average distance in pixel

	Δ = 1 bin	Δ = 1.5	avg. dist
Caltech all	86.1	94.4	24.8
Caltech tree 1	81.7	91.7	27.2
Caltech tree 2	95.0	96.7	24.7
Caltech tree 3	88.3	95.0	34.5
own db	96.1	98.9	11.7

5 Results

5.1 Position

A good object reference point is important for a precise object localization, so we first verify this. We list the performance of the position estimates with tolerance 1 and 1.5 times the bin width of the localization grid. When within the respective bound, the position estimate is considered correct, otherwise false.

The results in table 2 show that for the Caltech tree 2 category and our own leaves the position estimates are very good, with inferior results for the Caltech tree 1 and tree 3 categories. Inspecting the wrong estimates for the tree 1 category revealed a reason for this. The algorithm produces competing hypotheses for objects with partial rotation symmetric structures. A subset of the leaf "fingers" might hint at a different position (and orientation) of the leaf. For this specific dataset, mainly one leaf instance (with two main leaf peaks instead of one) was responsible for the errors. The algorithm chooses one of the two peaks as the main peak and estimates the position accordingly.

Such errors are easy to remedy, since we know the rough position and orientation of the object. We can, e.g., correlate an object mask in a small neighborhood of the estimated object position to refine the parameters. To verify that we only need to test a small vicinity, we increase the tolerance to 1.5 times the bin width. We can see that the performance increased dramatically. In Table 2 we list the average distance of the estimated to the real object reference point. For correct matches, the distance will usually be smaller, since its value is influenced by false results (estimated reference points not related to the object, usually far away in the image).

5.2 Orientation

We are not only capable of detecting the position of object class instances in an image. The core novelty in our

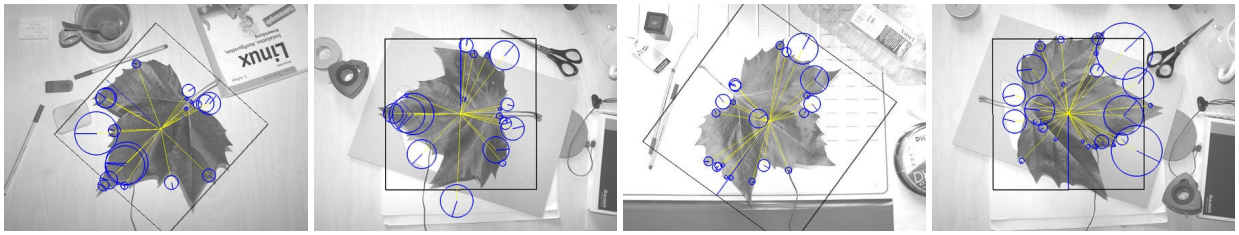

Figure 3: Example localization results on our database

Table 3: First column: correct orientation in %, other columns: Correct position and orientation in %

	$\Delta = \pi/12$	Δ = 1 bin	Δ = 1.5 bins
Caltech all	86.7	80.6	86.7
Caltech tree1	73.3	68.3	73.3
Caltech tree2	93.3	93.3	93.3
Caltech tree3	91.7	85.0	91.7
own db	94.3	92.3	94.3

approach is that we are also able to detect the orientation. Since we deal with natural objects, we allow a tolerance of $\pi/12$ in either direction for the rotation angle, for greater differences the estimated angle is considered false. The results for this experiment can be seen in table 3.

In spite of the results for almost all leaf categories being good, the orientation assignment for the Caltech tree 1 category seems worse. Again, the wrong estimates are mainly images from the outlier leaf instance with the two leaf peaks. Since for this leaf, no other images with this geometry can be found in the database, we consider this a plausible error. Please note that despite the less accurate position estimate for the Caltech leaves of type 3, the orientation was estimated correctly in most cases.

5.3 Localization

In the last test we show the total localization performance (position and orientation) of the approach, with the results listed in table 3. Images with either the reference point distance or orientation not in tolerance were considered false. Again, we list the results for distance threshold of 1 as well as 1.5 times the bin distance.

The results show that for geometrically stable objects, a very good overall performance can be achieved. Incorrect estimates are mainly due to a slightly inaccurate position hypothesis, orientation estimates were correct in most cases. This is especially visible when comparing the overall performance for the wider threshold with the orientation only evaluation: it is mostly the same for both experiments. As described above, the rough position hypotheses can be refined using the first position estimate as seed and, e.g., utilizing correlation to get a more exact match.

When looking at the results for the tests involving all Caltech leaf categories, we verify that the approach is capable of localizing variable leave instances with different leaf types in the reference database. The result for the whole database is only slightly worse than the average of the individual categories.

6 Conclusions

We introduced a method to localize rotated instances of visual object class members using local patch information. We could verify the capability to estimate the right position and angle for natural objects in cluttered scenes. Especially the orientation estimation works very well.

A prerequisite for the approach is that enough stable interest points can be found on an object. Here other scale invariant interest point detectors should also be tested, to further improve the results.

Acknowledgments

This work was partially funded by the EU MUSCLE NoE (FP6-507752). The segmentation software used in this work was written by Ilkka Luoma from PRIP, Vienna University of Technology. We would also like to thank Javier Cano for his KD-tree library.

References

[1] D. Ballard. Generalizing the Hough Transform to detect Arbitrary Shapes. *Pattern Recognition*, 13(2), 1981.

[2] G. Csurka, L. Dance, J. Willamowski, and C. Bray. Visual Categorization with Bags of Keypoints. In *ECCV Workshop on Stat. Learning in Comp. Vision*, pages 59–74, 2004.

[3] T. Deselaers, D. Keysers, and H. Ney. Discriminative training for object recognition using image patches. In *Proc. CVPR*, volume 2, pages 157–162, 2005.

[4] G. Dorko and C. Schmid. Selection of Scale-invariant Parts for Object Class Recognition. In *Proc. ICCV, 2003.*, pages 634–639 vol.1, 2003.

[5] L. Fei-Fei, R. Fergus, and P. Perona. Learning Generative Visual Models From Few Training Examples". In *Proc. of the Workshop on Generative-Model Based Vision*, Washington, DC, June 2004.

[6] R. Fergus, P. Perona, and A. Zisserman. A Sparse Object Category Model for Efficient Learning and Exhaustive Recognition. In *Proc. CVPR*, San Diego, June 2005.

[7] B. Leibe, A. Leonardis, and B. Schiele. Combined Object Categorization and Segmentation with an Implicit Shape Model. In *Proc. of the Workshop on Statistical Learning in Computer Vision*, Prague, Czech Republic, May 2004.

[8] D. G. Lowe. Distinctive Image Features from Scale-invariant Keypoints. *IJCV*, 60:91–110, 2004.

[9] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *Proc. CVPR*, 2006.