Semantic Object Recognition in Digital Images

Falk Schmidsberger and Frieder Stolzenburg
Hochschule Harz, Friedrichstr. 57–59
38855 Wernigerode, GERMANY
{fschmidsberger,fstolzenburg}@hs-harz.de

Abstract

In a digital image, each object is composed of segments with different shapes and colors. In order to recognize an object, e.g. a car or a book, it is necessary to find out, which segments are typical for this object and in which neighborhood of other segments they occur. If a typical segment in a characteristic neighborhood is found, this segment is part of the object to be recognized or the object itself. Typical adjacent segments for a certain object define the whole object in the image.

Following this idea, we introduce a procedure that learns typical segment configurations for a given object class by training with example images of the desired object. A hierarchical composition of segment clusters enables model building, taking into account the spatial relations of the segments in the image. The procedure employs methods from machine learning, namely clustering and decision trees, and from computer vision, e.g. image pyramid segmentation and contour signatures.

1 Introduction

Intelligent autonomous robots shall be able to identify objects in digital images, in order to navigate in their environment. To solve this task, we introduce a novel approach, combining methods from machine learning and computer vision, extending the predecessor paper [13].

To learn a new object class, a set of object images is provided for the recognition model training. Each image is split into its segments by color with image pyramid segmentation, to get suitable information for data mining. For each segment contour, a feature vector is computed that is invariant against rotation, scaling and translation. For this, we adopt three methods: polar distances, contour signatures, and ray distances. In order to reduce the number of feature vectors, a clustering method is used [3, 7] to build a cluster model. Each resulting cluster represents a set of similar feature vectors. In a second step, for all segments in one image, the segment clusters are determined with the cluster model and stored in a sample vector together with the object category of the image. This is done for all provided images. With these sample vectors, a decision tree model is trained [3, 7].

To predict an object class of an image, the image is split into its segments and, for all segments, the feature vectors are computed. With the trained cluster model and the feature vectors, the sample vector of the image is computed, and by means of the decision tree model, the object category of the image is predicted.

Related Works. The problem of recognizing and locating objects is very important in applications such as robotics and navigation. Therefore, there are numerous related works. The survey [5] reviews recent literature on both the 3D model building process

and techniques used to match and identify free-form objects from imagery, including recognition from 2D silhouettes. [6] describes shape surfaces by curves and patches, represented by linear primitives, such as points, lines, and planes. Results are presented for data obtained from a laser range finder. Hence, these results cannot be transferred directly to the analysis of video camera images, as done here. [9] presents an object recognition system that uses local image features, which are invariant to image scaling, translation, and rotation, and partially invariant to illumination changes and affine or 3D projection. This proposed model shares properties with the object recognition in primate vision. A nearest-neighbor indexing method is employed that identifies candidate object matches. [12] describes a model-based recognition system of planar objects based on a projectively invariant presentation of shape, using projective transformations. Index functions are used to select models from a model base, exploiting object libraries. However, for general semantic object recognition as considered here, fixed object libraries are certainly not sufficient.

2 The Approach

A digital image G can be represented as a two-dimensional point matrix. To extract a segment in an image each point (pixel) can be labeled. All pixels that make up a segment are given the same label [11]. The image is composed by a set of segments X_n after labeling [16], where the X_n are pairwise disjoint, forming a partition of G. Each object in a digital image is composed of a number of segments with different shapes and colors. In order to recognize an object, it is necessary to find out which segments are typical for which object and in which segment neighborhood they occur. If such a segment in a characteristic neighborhood is found, it is considered as part of the object. Typical adjacent segments for a certain object constitute the whole object in the image and allow its identification. The data mining methods clustering and decision trees are used to implement the approach. To process the segments of an image, a normalized feature vector is computed for each segment.

Segment Feature Vectors. The normalized feature vector V of a segment pixel set X (Fig. 1) comprises the data of three normalized distance histograms and is computed from the segment contour pixel set A (cf. Fig. 2) as $A = \{p \mid p \in X,\ p \text{ is contour point of } X\}$. A distance histogram (Fig. 3) consists of a vector, where each element contains the distance between the centroid s_X of the segment, i.e. the center of gravity, and a pixel in the segment contour or the distance between two pixels in the segment contour. These distance histograms are computed with the following three related methods: polar distance, contour signature and ray distance [1, 2, 8, 14].

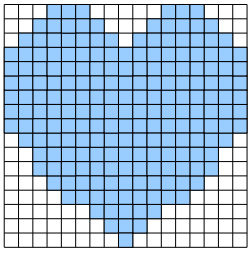

Figure 1: Segment example.

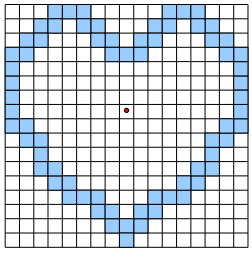

Figure 2: Segment contour A.

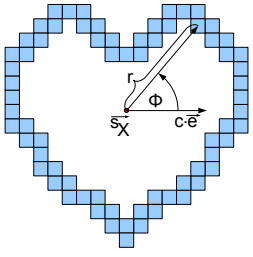

Figure 3: Polar distance r.

Polar Distance. Fixed angle steps of degree α with $0 < \alpha < 2\pi$, $\varphi = \alpha \cdot n$ and $n = 0, \ldots, \lceil 2\pi/\alpha \rceil - 1$ are used to select individual pixels in A with the maximum distance r to the centroid s_X of the segment (see Eq. 1 and Fig. 3). For non-convex segments, if there is no pixel with the actual angle φ, the pixel with the angle $\varphi + \pi$ and the minimum distance to s_X is chosen.

$$s_X = \begin{pmatrix} x_s \\ y_s \end{pmatrix} \ , \ x_s = \frac{1}{|X|} \sum_{i=1}^{|X|} x_i \ , \ y_s = \frac{1}{|X|} \sum_{i=1}^{|X|} y_i \quad (1)$$

For v_p, the vector from the centroid to a point p on the segment contour, it holds $v_p = s_X - p$. The angle φ of a contour point p around s_X is $\angle(v_p,\ e)$ with the unit vector $e = (1\ 0)^T$, and thus it holds $v_p \cdot e = |v_p| \cdot |e| \cdot \cos(\varphi_p)$. All selected pixels p are stored in the pixel set B (Fig. 4) and the distance r of each pixel to the centroid s_X is stored in the polar distance histogram vector MPD (maximum polar distance) with a constant number of elements for each segment.

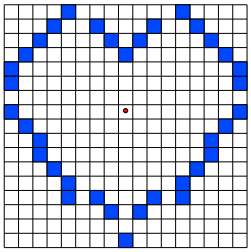

Figure 4: B with $\alpha = \frac{\pi}{18}$.

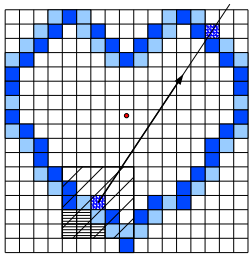

Figure 5: Contour signature.

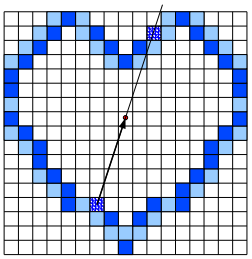

Figure 6: Ray distance.

Contour Signature. In the contour signature histogram vector, MCD (maximum contour distance), the distance d_{N_p} of each pixel in B to the corresponding opposite pixel in A is stored. In this case, the straight line between the two pixels has to have a $90°$ angle to the tangent through the actual pixel in B (cf. Fig. 5). The direction vector v_{CN} to the corresponding opposite pixel is approximated by the 24-neighborhood of the actual pixel p (Eq. 2, with n = 1 for the 24-neighborhood). This means, we consider a square of 5×5 pixels with p as midpoint. The corresponding opposite pixel $a \in A$ is the pixel with greatest distance to p on v_{CN}. MCD has the same cardinality as MPD.

$$v_{CN} = \sum_{x_q=x_p-1-n}^{x_p+1+n} \sum_{y_q=y_p-1-n}^{y_p+1+n} \begin{cases} p - \begin{pmatrix} x_q \\ y_q \end{pmatrix} & \forall n,q\ : q \notin X, n\ \in \mathbb{N} \text{ (fix)} \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases} \tag{2}$$

Ray Distance. In the ray distance histogram, the distance d_{C_p} of each Pixel in B to the corresponding pixel in A is stored as in Fig. 6. Here, the centroid s_X is on the straight line between the two pixels and the result is a distance histogram vector $MCCD$ (maximum center contour distance) with the same cardinality as MPD.

Feature Vector Normalization. In most cases, the distance histograms have different values even for the same segment, when this is rotated or resized (Fig. 7). To get a normalized segment feature vector, each distance histogram has to be normalized. At first, the rotation is normalized by smoothing the feature vector with a Gaussian filter and shifting the distance values of the vector, so that the angle with the maximum value and the maximum angle difference to the next angle with the maximum value is the first element in the feature vector (Fig. 8). In a second step, the values itself are normalized to $[0.0, 1.0]$, by dividing all distance values by the respective maximum distance value (Fig. 9). After the normalization, all three distance vectors are joined to the new feature vector V of the segment which is invariant against translation, rotation and resizing.

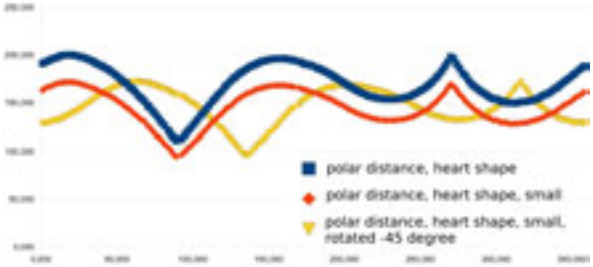

Figure 7: Polar distances of three heart shapes.

Clustering. In order to reduce the number of feature vectors, a clustering algorithm is used to build a cluster model [3, 7]. Each resulting cluster represents a set of similar feature vectors, identified by its respective centroid. Fig. 10 (upper half) shows the centroid of one cluster of histograms for the three methods and their polar representations

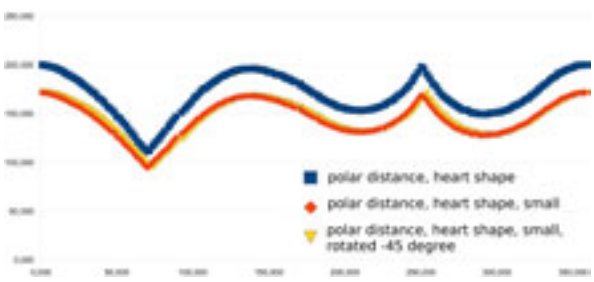

Figure 8: Polar distances with normalized rotation.

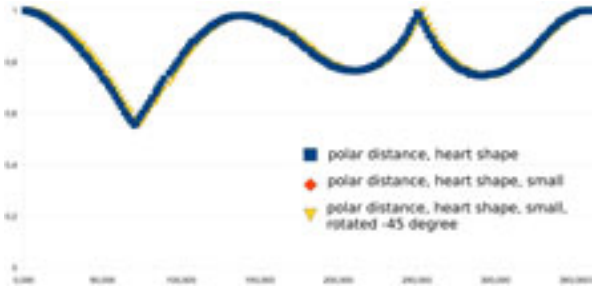

Figure 9: Polar distance, normalized rotation and values.

(lower half) for the running example (cf. Fig. 1), where all distance lines are drawn in polar representation, i.e. around a center point. Fig. 11 shows a trained cluster model. Fig. 10 is one cluster in it, namely the last one. The size of the circle in each subdiagram gives a measure for the variance of the histograms in the respective cluster. The background color indicates the number of vectors forming this cluster. The cluster model can now be used to decide the cluster affiliation for a new given feature vector.

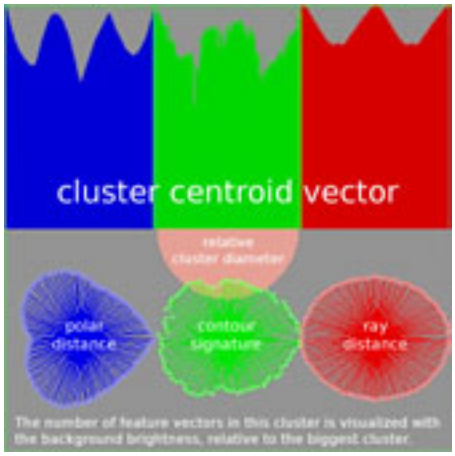

Figure 10: Cluster visualization.

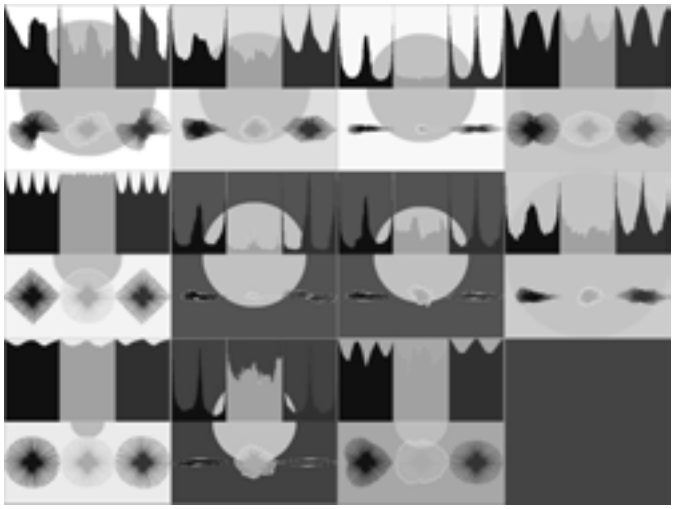

Figure 11: Cluster model visualization (11 clusters).

Decision Trees. For all segments in one image, the cluster numbers for each segment are computed and stored in a sample vector together with the object category of the image. This is done for all training images. With this sample vectors, a decision tree model is trained [3, 7]. Finally, the trained decision tree model is used to decide which object is described by the given sample vector.

Implementation. The presented algorithms were implemented in the programming language C++ using the OpenCV library [10]. To get the segments of the digital images, an image pyramid segmentation algorithm in OpenCV is employed [4]. The clustering of the feature vectors is done with OpenCV and the cluster model building has been implemented additionally. The OpenCV decision tree model implementation was used to learn the object classification with the sample vectors.

3 Application

The Semantic Robot Vision Challenge. The implemented algorithms were tested in a challenging field of application. In the Semantic Robot Vision Challenge 2009 [15], a robot had 2 hours to find image examples on the Internet and to learn visual models for 20 objects, given as a text list. After that, the objects have to be identified in the indoor environment within 30 minutes without an Internet connection (45 images

were provided in the software league). During the challenge, one image was classified correctly, 14 images were falsely classified and for the remaining 30 images no category was found. On 9 images was no classifiable object. In detail, the procedure and implementation is described in [13].

Image Datasets. To test and improve the implemented algorithms in a controlled environment, they were used to classify images from the butterfly image dataset (Fig. 12) [17]. Compared to the implementation above, the image preprocessing is improved. Smoothed distance histograms and a clustering algorithm with an automatically adapted cluster count for each image category result in cluster models with less but more precise clusters. One decision tree model for each image category is trained with one sample vector for each image structured as follows: The first k entries contain the number of found segments associated to the respective cluster and the entry $k+1$ is the given category identifier of the image.

To evaluate the implementation, the image dataset with two categories is segmented in a training set and a test set of images. The models are only trained with the images from the training set. The trained models are used to predict the right category for a given image, with a success rate of 94.44 % if the image is from the training set and a success rate of 60.00 % if the image is from the test set.

Figure 12: Butterfly image dataset example.

Figure 13: Visualization of a segment tree.

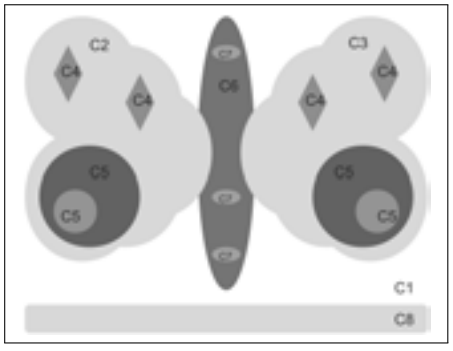

Figure 14: Segment cluster assignments.

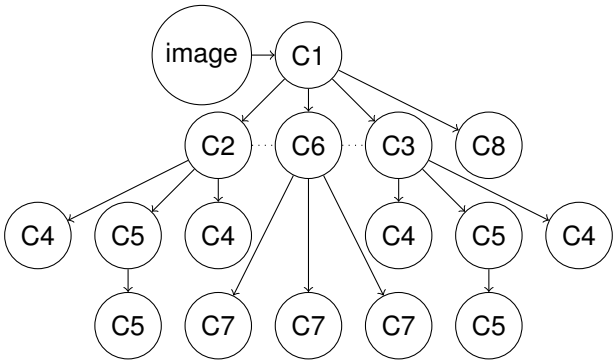

Figure 15: Segment Cluster Tree.

One idea to get better object recognition rates is using spatial relations between segments in the image. For this, we extended our approach using different kinds of views onto the segments by sorting the segment clusters in a hierarchy tree. Each node in the tree represents a specific segment cluster. The root node represents the image itself. A child node represents a segment which is immediate part of the segment of the upper level (cf. Fig. 14 and 15). Nodes on the same level are marked as neighbors by dotted lines, if the corresponding segments in the image are connected. Fig. 13 shows the visualization of the segment tree computed from the original image (Fig. 12). Different colors mean different levels in the segment hierarchy.

We can now extract 4 different types of sample vectors from the data: The first one contains all paths from a leaf node to the root, the second one all child nodes of a node one level above, the third one all nodes marked together as neighbors, and the fourth one all child nodes of all nodes in the tree. These five different sample vector types are used to train five decision tree models, which are combined to predict the right object category. The success rate is 100.00 % if the image is from the training set and 67.50 % if the image is from the test set.

4 Future Work

Our first results are encouraging, but in the future, the implementation of our approach will be improved further to become faster with an increased object recognition success rate. For this, a distributed implementation seems to be promising. Using more spatial relations of the segments for a more accurate decision tree model and using another decision tree techniques like AdaBoost is also desirable. The final goal of the project is to implement the approach as a real-time object recognition system feasible for autonomous multi-copters, i.e. flying robots with several propellers.

References

[1] E. Alegre, R. Alaiz-Rodríguez, J. Barreiro, and J. Ruiz. Use of contour signatures and classification methods to optimize the tool life in metal machining. *Estonian Journal of Engineering*, 1:3–12, 2009.

[2] H. Bässmann and J. Kreyss. *Bildverarbeitung Ad Oculos*. Springer, 4th edition, 2004.

[3] M. J. A. Berry and G. Linoff. *Data Mining: Techniques For Marketing, Sales, and Customer Support*. John Wiley & Sons Inc., 1997.

[4] G. Bradski and A. Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media Inc., 2008.

[5] R. J. Campbell and P. J. Flynn. A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding*, 81(2):166 – 210, 2001.

[6] O. Faugeras and M. Hebert. The representation, recognition, and locating of 3-D objects. *The International Journal of Robotics Research*, 5(3):27–52, 1986.

[7] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufman Publishers, 2nd edition, 2006.

[8] B. Jähne. *Digital Image Processing*. Springer, 6th revised and extended edition, 2005.

[9] D. G. Lowe. Object recognition from local scale-invariant features. *Computer Vision, IEEE International Conference on*, 2:1150, 1999.

[10] OpenCV (open source computer vision) library. http://opencv.willowgarage.com/wiki/, 2010.

[11] M. Petrou and P. Bosdogianni. *Image Processing. The Fundamentals*. John Wiley & Sons Ltd, 1999.

[12] C. A. Rothwell, A. Zisserman, D. A. Forsyth, and J. L. Mundy. Planar object recognition using projective shape representation. *International Journal of Computer Vision*, 16:57–99, 1995.

[13] F. Schmidsberger and F. Stolzenburg. Semantic object recognition using clustering and decision trees. In J. Filipe and A. Fred, editors, *Proceedings of 3rd International Conference on Agents and Artificial Intelligence*, volume 1, pages 670–673, Rome, Italy, 2011.

[14] F. Shuang. Shape representation and retrieval using distance histograms. Technical report, Dept. of Computing Science, University of Alberta, 2001.

[15] Semantic robot vision challenge. http://www.semantic-robot-vision-challenge.org, nov 2009.

[16] J. Steinmüller. *Bildanalyse. Von der Bildverarbeitung zur räumlichen Interpretation von Bildern*. Springer, 2008.

[17] C. S. Svetlana Lazebnik and J. Ponce. Semi-local affine parts for object recognition. In *Proceedings of the British Machine Vision Conference*, volume 2, pages 959–968, 2004.