SEMANTIC OBJECT RECOGNITION USING CLUSTERING AND DECISION TREES

Falk Schmidsberger and Frieder Stolzenburg
Dep. of Automation and Computer Sciences, Hochschule Harz, Friedrichstr. 57–59, 38855 Wernigerode, Germany

Keywords: Vision and perception, Data mining, Clustering, Decision trees, Object recognition, Image understanding, Autonomous robots.

Abstract: Each object in a digital image is composed of many patches (segments) with different shapes and colors. In order to recognize an object, e.g. a table or a book, it is necessary to find out which segments are typical for which object and in which segment neighborhood they occur. If a typical segment in a characteristic neighborhood is found, this segment will be part of the object to be recognized. 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, which can be found in and downloaded from the Internet. The procedure employs methods from machine learning, namely *k*-means clustering and decision trees, and from computer vision, e.g. contour signatures.

1 INTRODUCTION

Intelligent autonomous robots have to identify objects in digital images, in order to navigate in their environment. To solve this task, we introduce a new approach in this paper, combining methods from machine learning and computer vision. It consists of a training and an analysis phase.

The training phase consists of two major steps: In the first step, all downloaded training images are split into their segments by color. 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 *k*-means clustering method is used (Berry and Linoff, 1997; Han and Kamber, 2006). Each resulting cluster represents a set of similar feature vectors.

In the second step, for all segments in one image, the clusters for each segment and its adjacent segments are determined and stored in a sample vector together with the object category of the image. Segments are considered adjacent if parts of their contour coincide. This is done for all downloaded training images. With these sample vectors, a decision tree model is trained (Berry and Linoff, 1997; Han and Kamber, 2006).

In the analysis phase, each provided image is split into its segments by color, and for all these segments, the feature vector is computed. Each segment that could not be recognized by the cluster model is ignored. For all remaining segments, the sample vector including the adjacent segments is computed, and by means of the decision tree model, the object category is predicted. All the adjacent segments with the same predicted object category are composed to a compound segment. Each of these compound segments represents an object in the image.

The selection of one image for each object category is the last step of the program. The image with the biggest number of segments in a compound segment with the right object category is selected.

2 THE APPROACH

A digital image G can be represented as a two-dimensional point matrix and composed by a set of segments X_n (see Eq. 1, cf. Steinmüller, 2008).

$$G = \bigcup_{n=1}^{N} X_n \quad \text{with} \quad X_{n_1} \cap X_{n_2} = \emptyset \tag{1}$$

Each object in a digital image is composed of a number of segments with different shapes and colors. 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 will be part of the object. Typical adjacent segments

Schmidsberger F. and Stolzenburg F..
SEMANTIC OBJECT RECOGNITION USING CLUSTERING AND DECISION TREES.
DOI: 10.5220/0003188706700673
In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence* (ICAART-2011), pages 670-673
ISBN: 978-989-8425-40-9

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.

2.1 Normalized Segment Feature Vector

The normalized feature vector V of a segment X (Fig. 1) comprises the data of three normalized distance histograms and is computed from the segment contour A (cf. Fig. 2) as follows:

$$A = \{p \mid p \in X,\ p \text{ is contour point of } X\} \tag{2}$$

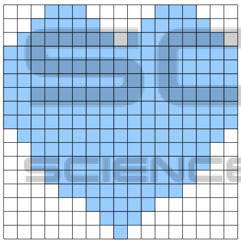

Figure 1: Segment example.

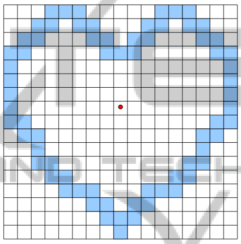

Figure 2: Segment contour A.

A distance histogram consists of a vector, where each element contains the distance between the centroid s_X of the segment, i.e. the center of gravity (Fig. 3), 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 (Alegre et al., 2009; Jähne, 2005; Bässmann and Kreyss, 2004; Shuang, 2001). We explain them briefly in the next few sections.

2.1.1 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. 3 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. It holds:

$$s_X = \begin{pmatrix} x_s \\ y_s \end{pmatrix} \ , x_s = \frac{1}{|X|} \sum_{i=1}^{|X|} x_i \ , \ y_s = \frac{1}{|X|} \sum_{i=1}^{|X|} y_i \tag{3}$$

$$v_p = s_X - p \tag{4}$$

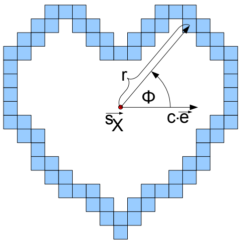

Figure 3: Polar distance r.

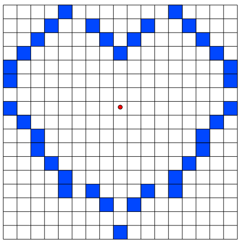

Figure 4: Pixel set B selected by the polar distance, $\alpha = \frac{\pi}{18}$.

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:

$$v_p \cdot e = |v_p| \cdot |e| \cdot \cos(\varphi_p) \tag{5}$$

All selected pixels 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.

2.1.2 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 degree angle to the tangent through the actual pixel in B.

The direction vector v_{CN} to the corresponding opposite pixel is approximated by the 24-neighborhood of the actual pixel p (Fig. 5, Eq. 6, 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 biggest distance to p on v_{CN}. *MCD* has the same cardinality as *MPD*.

$$v_{CN} = \sum_{x_q = x_p - 1 - n}^{x_p + 1 + n} \ \sum_{y_q = y_p - 1 - n}^{y_p + 1 + n} \begin{cases} p - \begin{pmatrix} x_q \\ y_q \end{pmatrix} & \forall n, q : \begin{array}{r} q \notin X, \\ n \in \mathbb{N} \text{ (fix)} \end{array} \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases} \tag{6}$$

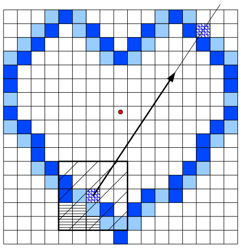

Figure 5: Contour signature.

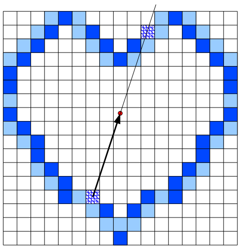

Figure 6: Ray distance.

2.1.3 Ray Distance

In the ray distance histogram, the distance d_{C_p} of each Pixel in B to the corresponding pixel in A like in Fig. 6 is stored. 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*.

2.1.4 Histogram Normalization

In most cases, the distance histograms have different values even for the same segment, when this is rotated or resized (Fig. 7).

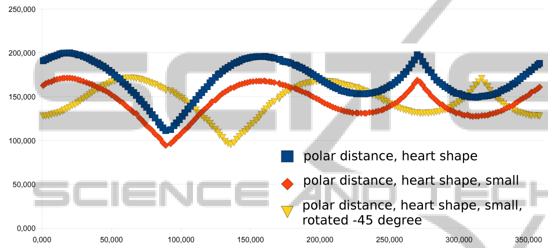

Figure 7: Polar distances of three heart shapes.

To get a normalized segment feature vector, each distance histogram has to be normalized. At first, the rotation is normalized (Fig. 8).

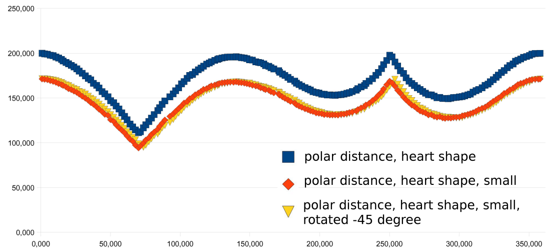

Figure 8: Polar distances with normalized rotation.

In a second step, the values itself are normalized to the range between 0.0 and 1.0, by dividing the original distance values by the respective maximum distance value (Fig. 9).

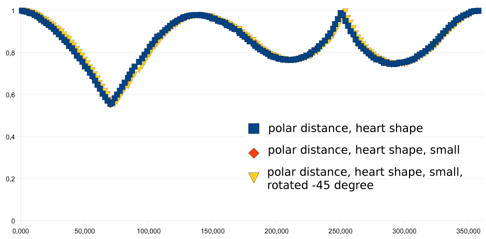

Figure 9: Polar distances with normalized rotation and size.

After the normalization, all three distance vectors will joined. Now the feature vector V of the segment is invariant against translation, rotation and resizing.

2.2 Clustering

In order to reduce the number of feature vectors, a k-means clustering algorithm is used to build a cluster model (Berry and Linoff, 1997; Han and Kamber, 2006). Each resulting cluster represents a set of similar feature vectors, and the trained cluster model can be used to decide the cluster affiliation for a new given feature vector.

2.3 Decision Trees

For all segments in one image, the clusters for each segment and its adjacent segments are computed and stored in a sample vector together with the object category of the image. This is done for all downloaded training images. With this sample vectors a decision tree model is trained (Berry and Linoff, 1997; Han and Kamber, 2006). Finally, the trained decision tree model is used to decide which object is described by the given sample vector.

3 APPLICATION

3.1 Semantic Robot Vision Challenge

To test the algorithms in a challenging field of application, they were implemented for the Semantic Robot Vision Challenge 2009 (SRVC, 2009).

In this challenge, a robot has 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 environment within 30 minutes without an Internet connection (45 images were provided in the software league).

3.1.1 Implementation

The presented algorithms were implemented in the programming language C++ using the OpenCV library (OpenCV, 2010).

To get the segments of the digital images, an image pyramid segmentation algorithm in OpenCV is employed (Bradski and Kaehler, 2008). The computation of the contours and the segment feature vectors is implemented by the first author. The k-means clustering of the feature vectors can be done with OpenCV, but building the cluster model has been implemented by the first author, additionally. The OpenCV decision tree model implementation was used to learn the object classification with the sample vectors.

3.1.2 Processing the Data and Evaluation

In detail, the concretely implemented procedure works as follows. Here, all constants are experimentally determined to train the models in less than the given 120 minutes and to classify the provided images in less than 30 minutes.

Step 1: Training. Up to 25 images were downloaded from the Internet for each object on the list. All downloaded images were segmented by color and for each resulting segment, 39083 segments altogether, a feature vector V with 300 entries was computed (cardinality of MPD, MCD and $MCCD = 100$). After the association of the feature vectors to 1000 clusters with k-means clustering, the cluster model is build from the cluster associations.

Using the cluster model the decision tree model is trained with a sample vector for each segment structured as follows: Each sample vector has $k+2$ entries (i.e. chosen cluster count $+2$). The first k entries contain the number of segments associated to the respective cluster in the neighborhood of the actual segment of the image. In this context, neighborhood means that the bounding boxes of the segments overlap or have a distance less than 3 pixels. The entry $k+1$ contains the cluster number of the actual segment and the value of the entry $k+2$ is the category identifier of the actual category of the image.

Step 2: Classification of the Unknown Images. For each segment in the image the feature vectors V and the sample vectors are created (without the category of the image). The decision tree model predicts the image category with the sample vectors. Each predicted category of the image and the number of segments in the neighborhood of the actual segment is stored. The category with the most number of segments in the neighborhood is chosen as the category of the image.

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 there was not any classifiable object).

4 FUTURE WORK

Our first results are encouraging, but in the future, the implementation of our approach has to be faster with an increased object recognition success rate.

For that, the image preprocessing and the segmentation algorithm have to be improved, in order to support a better classification. Smoothing the distance histograms to reduce measurement artifacts, using a clustering algorithm with a variable cluster count to get a cluster model with less but more precise clusters and using more spatial relations of the segments for a more accurate decision tree model is also desirable.

The goal 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

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

Bässmann, H. and Kreyss, J. (2004). *Bildverarbeitung Ad Oculos*. Springer, Berlin, Heidelberg, New York, 4th edition.

Berry, M. J. A. and Linoff, G. (1997). *Data Mining: Techniques For Marketing, Sales, and Customer Support*. John Wiley & Sons Inc., New York, Chichester, Weinheim, Brisbane, Singapore, Toronto.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media Inc., Beijing, Cambridge, Farnham, Köln, Sebastopol, Taipei, Tokyo.

Han, J. and Kamber, M. (2006). *Data Mining: Concepts and Techniques*. Morgan Kaufman Publishers, Amsterdam, Boston, Heidelberg, London, New York, Oxford, Paris, San Diego, San Francisco, Singapore, Sydney, Tokyo, 2nd edition.

Jähne, B. (2005). *Digitale Bildverarbeitung*. Springer, Berlin, Heidelberg, New York, 6th edition.

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

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

SRVC (2009). Semantic robot vision challenge. http://www.semantic-robot-vision-challenge.org.

Steinmüller, J. (2008). *Bildanalyse. Von der Bildverarbeitung zur räumlichen Interpretation von Bildern*. Springer, Berlin, Heidelberg.