Object Recognition Using Hough-transform Clustering of SURF Features

Viktor Seib, Michael Kusenbach, Susanne Thierfelder, and Dietrich Paulus

Active Vision Group (AGAS), University of Koblenz-Landau
Universitätsstr. 1, 56070 Koblenz, Germany
{vseib, mkusenbach, thierfelder, paulus}@uni-koblenz.de
http://robots.uni-koblenz.de

Abstract. This article describes the object recognition approach used by team homer@UniKoblenz in the RoboCup@Home league. It is based on feature extraction from rgb scene images. Features are matched with features in learned object models and clustered in Hough-space to find a consitent object pose. Using this approach, team homer@UniKoblenz won the Technical Challenge of the RoboCup@Home league in 2012. The described approach is available online as open source software provided as a ROS package: `http://wiki.ros.org/agas-ros-pkg`

Keywords: Object Recognition, Generelized Hough-transform, Hough-transform clustering, SURF

1 Introduction

We describe the object recognition approach used by team homer@UniKoblenz in the RoboCup@Home league. This algorithm has been designed to work with partially occluded objects, cluttered background, change in ullimination and arbitrary object poses.

As feature detector and descriptor we use SURF [1]. SURF is a point feature detector which also provides a descriptor for matching. Its main advantage is the fast computation while the features are distinctive enough to enable robust object recognition even under difficult circumstances such as partial occlusion and cluttered background.

The descriptors from the extracted interest points are matched with objects from a database of features using nearest-neighbor matching. The identification of clusters for a certain object was accomplished by using Hough-transform clustering to abtain valid object pose hypotheses. In the final step a homography is built using the clustered features to verify consistent pose parameters and select the most probable object pose for each recognized object.

2 Hough-Transform Clustering of SURF Features

Our object recognition approach is based on 2D camera images. In the training phase, SURF features are extracted and saved in a database. The recognition

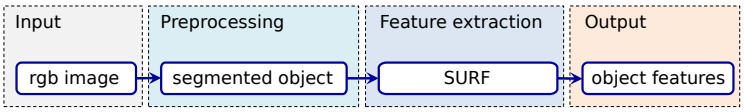

Fig. 1. Image processing pipeling for the training phase

phase calculates SURF features on an input image and matches features with the database using nearest-neighbor matching. Wrong correspondences are removed and object pose transformations are clustered in a multi dimensional Hough-space accumulator. Finally, the matched features are verified by calculating a homography.

To sum up, we extract from each image a number of SURF features f. A feature is a tuple f = (x, y, σ, θ, δ) containing the position (x, y), scale σ and orientation θ of the feature in the image, as well as a descriptor δ. Thus, the features are invariant towards scaling, position in the image and in-plane rotations. They are also robust towards changes in illumination and lesser off-plane rotations.

2.1 Training

The image processing pipeling for the training phase is shown in Fig. 1. In order to train the object recognition classificator an image of the background has to be captured. Subsequently, an image of the object is acquired. From this two images a difference image is computed to separate the desired object from the background. Depending on the light conditions, the object might cast a shadow on its surroundings. Naturally, this shadow would appear in the difference image and thus be considered as part of the object. Therefore, the borders of the extracted object can be adjusted to reduce the area contributed to the object and thus remove shadows from the foreground. From the acquired object image SURF features are extracted and stored in an object database. Further, images with a different object view can be acquired and added to the object model in the database. In their original publication, Bay et al. recommend 30° as an optimal rotation between subsequently acquired images of an object for SURF computation [1]. It is not necessary to acquire different rotations of the same object view, since SURF features and the presented algorithm are rotation invariant.

2.2 Recognition

The image processing pipeling for the recognition phase is shown in Fig. 2. During object recognition no information about the background is available. Thus, SURF features are computed on the whole input image. The obtained features are then matched against the features stored in the object database using nearest neighbor matching.

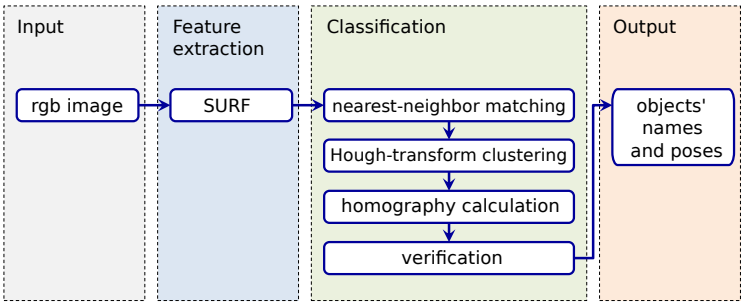

Fig. 2. Image processing pipeling for the recognition phase

Nearest-Neighbor Matching For each feature in the input image the best feature in the database is obtained by calculating a distance measure based on the euclidean distance of the feature descriptors. Since simple distance thresholds do not perform well in high dimensional space, Lowe introduced the distance ratio [5], which is used here. The distance ratio is the ratio of the euclidean distance to the best fitting and the second best fitting descriptor. If this quotient is low enough, the best fit is considered a matching feature. If the quotient is higher than a given threshold, the best and the second best descriptor fit almost equally well. This leads to the assumption that they are very likely the best matches only by chance and not because one of them actually matches the query descriptor. The distance ratio also sorts out ambiguous matches, which may result from repeated textures on objects. For the fast nearest-neighbor and distance computation in the high dimensional descriptor space we use an approximate nearest neighbor approach [6].

Since SURF features are calculated on the whole input image, including a potentially different background than in the training phase, not all features are matched in this step. The result of feature matching is a set of matches between features extracted from training images and the scene image. This set may still contain outliers, i. e. matches between features which do not correspond to the same object point. These erroneous correspondences are discarded in the next step.

Hough-transform Clustering Each feature match gives a hint of the object's pose in the scene image. To cluster this information from all feature matches, a four dimensional Hough space over possible object positions (x, y, σ, θ) is created. Here, (x, y) is the position of the object's centroid in the image, σ it's scale, and θ it's rotation. The goal is to find a consistent object pose in order to eliminate false feature correspondences from the previous step. This four dimensional accumulator is visualized in Fig. 3. The red boxes represent translation in x- and y-directions. Inside each red box, the x-axis represents scale and the y-axis represents rotation. Each pixel inside a red box corresponds to a bin.

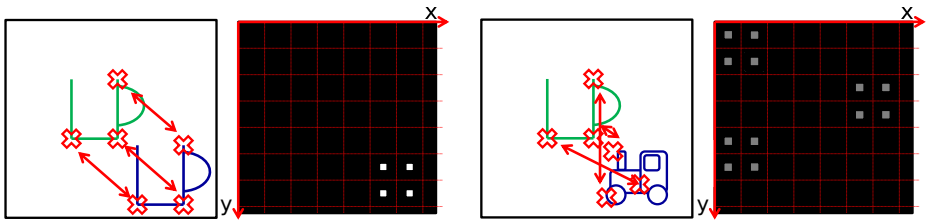

Fig. 3. Objects are recognized and hypotheses about the object pose are clustered in Hough-space using the four dimensional accumulator. Maxima appear white (left), outliers appear gray (right). Red boxes indicate the translation in x- and y-direction. Inside each red box horizontal and vertical translation encode the scale and rotation, respectively.

Each feature correspondence is a hypothesis for an object pose and is added to the corresponding bin in the accumulator. As suggested in [5] and [3], to reduce discretization errors, each hypothesis is added to the two closest bins in each dimension, thus resulting in 16 accumulator entries per feature correspondence. Clusters of maxima in the Hough-space correspond to most probable object poses, whereas bins with erroneous object poses get only little votes (Fig. 3). Thus, outliers are removed and correct object poses are obtained. For the next steps only bins with at least five entries are considered, since we want to find a consistent pose applying homography calculation. This low threshold helps finding small as well as partially occluded objects in the scene.

We chose 10 bins for each dimension in the Hough-space, resulting in 10^4 bins describing different possible object positions in the image. Each feature correspondence votes into 16 bins (the one it falls into and the closest ones of each dimension to avoid discretization effects). More bins per dimension would allow for a finer quantization of feature poses. However, this would also lead to potential maxima being scattered among neighboring bins. Objects with sufficient feature correspondences would be recognized with less confidence or would not be recognized at all. Choosing less bins on the other hand would lead to feature correspondences voting for wrong object poses or even wrong objects.

To calculate the bin for the object position the centroid of the object in the scene s_c has to be estimated. In the following, values o describe properties of an object acquired during the training phase, whereas values s refer to keypoints found in a test scene during the recognition phase. The translation vector $\mathbf{v}'$ from the centroid of the learned object $\mathbf{o}_c$ to the position of the feature keypoint $\mathbf{o}_p$ in the object, normalized with the scale ratio of the scene keypoint s_σ and the object keypoint o_σ is calculated according to Eq. 1.

$$\mathbf{v}' = (\mathbf{o}_c - \mathbf{o}_p)\frac{s_\sigma}{o_\sigma} \tag{1}$$

The resulting vector v' has to be rotated to account for possible object rotation in the scene. This is done by applying Eq. 2

$$\mathbf{v} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \mathbf{v}' \tag{2}$$

where $\alpha = |o_\theta - s_\theta|$ is the minimal rotation angle between the feature rotation o_θ in the object and the feature rotation s_θ in the scene. Finally, the estimated centroid of the object in the scene s_c is obtained by adding the feature's position in the scene $\mathbf{s}_p$ to the calculated translation vector $\mathbf{v}$ (Eq. 3).

$$\mathbf{s}_c = (s_{cx}, s_{cy})^T = \mathbf{v} + \mathbf{s}_p \tag{3}$$

The bins for the x- and y-location in the accumulator, i_x and i_y, are calculated as shown in Eq. 4 and Eq. 5,

$$i_x = \left\lfloor \frac{s_{cx}}{w} b_x \right\rfloor \tag{4}$$

$$i_y = \left\lfloor \frac{s_{cy}}{h} b_y \right\rfloor \tag{5}$$

where w and h refer to the image width and height, and b_x and b_y is the number of bins in the x- and y-dimension of the Hough-accumulator, respectively. Apart from i_x and i_y also the corresponding bins with position indices $i_x + 1$ and $i_y + 1$ are incremented to reduce discretization errors.

The index for the rotation bin is calculated using the difference between the angles of the key point correspondences α and the total number of bins reserved for the rotation in the accumulator b_r. Originally, α is in $[-\pi, \pi]$, thus Eq. 6 normalizes the angle to be in $[0, 2\pi]$.

$$i'_r = \frac{(\alpha + \pi) b_r}{2\pi} \tag{6}$$

To allow for the rotations by $-\pi$ and π to be close together in the accumulator the final index for the rotation bin is calculated according to Eq. 7.

$$i_r = \lfloor i'_r \rfloor \mathrm{mod} b_r \tag{7}$$

Again, a second bin is used to reduce discretization errors. It's index is calculated according to Eq. 8:

$$i_r = \lfloor i'_r + 1 \rfloor \mathrm{mod} b_r. \tag{8}$$

The fourth dimension in the accumulator encodes the scale the point of interest was found at. To determine the accumulator bin for scale, first the ratio q between the scales of the key point in the scene s_σ and in the learned object o_σ is needed (Eq. 9):

$$q = \frac{s_\sigma}{o_\sigma}. \tag{9}$$

Further, the index is determined by the total number of bins used to represent scale b_s and the number of octaves n used for SURF extraction and is calculated according to Eq. 10:

$$b_s = \left\lfloor \frac{\log_2(q)}{2(n-1)} + 0.5 \right\rfloor b_s. \tag{10}$$

As before, discretization errors are reduced by using a second bin with the index $b_s + 1$. All scales that go beyond the range represented by the last bin are subsumed in the bin for the biggest scale of the accumulator.

As a result of the Hough-transform clustering all features with consistent poses are sorted into bins, while most outliers are removed because they don't form maxima in Hough-space (Fig. 3). So far, all features were processed independently of all other features without taking into account the geometry of the whole object. With the resulting possible object poses from the Hough-transform clustering, the goal in the next step is to find the best geometric match with all features in one accumulator bin.

Homography Calculation In this step, bins representing maxima in Hough-space are inspected in order to find the bin that matches best the object pose. All bins containing five keypoint correspondences or more are considered as maxima. A perspective transformation is calculated between the features of a bin and the corresponding points in the database under the assumption that all features lie on a 2D plane. As most outliers were removed by discarding minima in Hough-space, a consistent transformation is obtained here. RANSAC is used to identify the best homography for the set of correspondences. The homography with most point correspondences is considered to be the correct object pose. Using the obtained homography the recognized object can be project into the scene (Fig. 8). Since homography calculation is computationally expensive the runtime of the object recognition algorithm would increase considerably if a homography was calculated for each bin. To speed up the algorithm all bins are sorted in descending order considering the number of features. A homography is calculated starting with the bin containing the highest number of features. The calculation terminates, if the next bin contains less features than the number of found point correspondences in the calculation of the best homography so far. The result is a homography describing the relative position, orientation and scale of the best fitting training image for a test image, as well as the number of features supporting this hypothesis.

Verification of Results The last step of our object recognition pipeline verifies the results. Using a threshold of a minimal matched feature number to verify the presence of an object in the scene is futile since large and heterogeneously structured objects contain more features than small and homogeneously structured objects. Instead, an object presence probability p is calculated as

$$p = \frac{f_m}{f_t} \tag{11}$$

where f_m is the number of matched features of that object and f_t is the total number of features that are present in the area of the object. The number of

features in the object area is calculated by projecting the object into the scene using the homography and then counting all features in the bounding box of the projected object.

3 Evaluation

This Section describes the different experiments performed to evaluate the presented object recognition approach. Experiments were performed to test the influence of the accumulator size, variable background and light conditions, as well as partial occlusion on the performance of the classification.

For the verification step we used a threshold of 15 % (Eq. 11) and a minimum of 5 matched features per object. These two values have the most influence on the number of false positive recognitions. If they are not chosen restrictively enough, the number of false positive recognitions increases. On the other hand, if they are chosen too restrictively, no objects would be recognized or a higher number of training views per object would be required to provide enough features for matching.

When not stated otherwise, the accumulator size is 10 bins in each dimension. For the evaluation, objects from the database presented in [4] were used. All images in this database have a resolution of 640×480 pixels.

3.1 Performance

The evaluation was performed on an off-the-shelf notebook with an Intel Core 2 processor with 2 GHz and 2048 MB RAM. We measured the processing time of the algorithm for one object view and a scene image with weak heterogeneous background and the learned object in the same pose. The object view used in this test is depicted in Fig. 4. The processing time needed for each step of our object recognition algorithm is presented in Tab. 1

Initially, 500 key points are detected in the scene image. The keypoint detection step takes the most time, since key points are not only extracted from the object, but also from the background. The nearest neighbor matching yields 139 key point correspondences between the scene image and the learned object view. However, some of these correspondences are erroneous as some of them

Table 1. Calculation time for each algorithm step

Algorithm Step	# Key Points	Time [ms]
Detection	500	697
NN-Matching	139	61
Hough-clustering	102	13
Homography	98	12

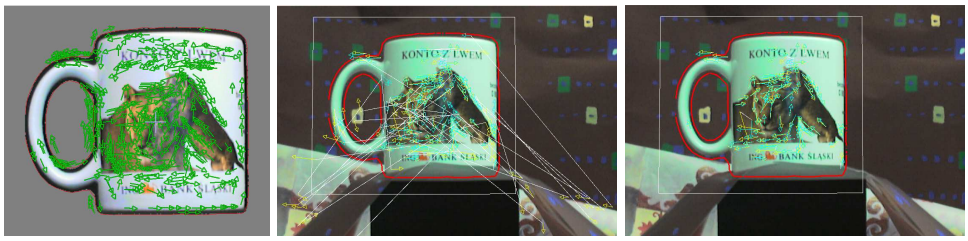

Fig. 4. Left: Object view used for time measurement. Green arrows indicate key points. Center: Key point correspondences after nearest neighbor matching. Some key points of the learned object view are matched to the background. Right: Recognition result after eliminating erroneous correspondences.

are matched with the background (Fig. 4). After the Hough-transform clustering only 102 correspondences remain. Finally, after calculating two homographies in 12 ms the best is found with 98 remaining correspondences to form the object recognition result (Fig. 4).

A total of 783 ms is needed for the calculation. This time increases if multiple object and object views are loaded into the object database, as the extracted features have to be compared with all data in the database. It is therefore crucial not to learn too many views of an object (see next Subsection). However, the most time consuming step (key point extraction) has to be performed only once per scene image.

3.2 Learning Object Views

We performed an experiment to determine how many object views are necessary for reliable object recognition. In order to do this it has to be determined by what angle an object can be rotated without loosing too many key points. For the evaluation, a single object view was acquired. Without loss of generality the object view was defined as depicting a pose with a rotation of 180° about its vertical axis. Subsequently, images from the database showing the object at different rotations were used for testing. As shown in Fig. 5 the number of matched features decreases rapidly for rotations beneath 150° and above 220°. Thus, a rotation of 30° between subsequently acquired image views is a good trade-off between number of found features and image views.

3.3 Accumulator Size

The size of each dimension of the accumulator is a crucial parameter for the performance of the algorithm. In our approach 10 bins per dimension proved to be a good trade-off between quantization errors (if too many bins are used) and insufficient accuracy (if too little bins are used). More than 10 bins lead to a shorter runtime as the features are distributed among a greater bin number, thus leading to less bins with a sufficiently large number of features for further

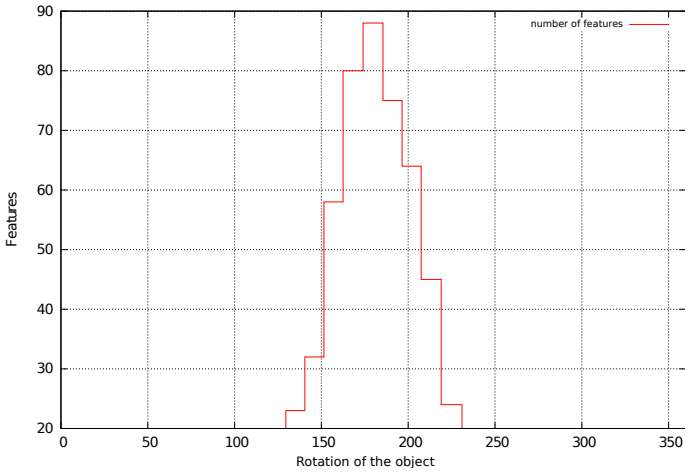

Fig. 5. Number of matched features of an object view depending on the object's rotation. The object view was acquired at a rotation of 180°.

processing. However, at the same time less features can be matched leading to unreliable recognition results.

3.4 Background Variation

The experimental results of our algorithm with different backgrounds are presented in Tab. 2. A comparison of our algorithm with a statistical object recognition approach was given in [2]. The algorithm was trained with 5 different objects (Fig. 6) and 5 views per object from [4]. The classification was performed on the same 5 objects, but with 10 different views per object. The employed database contains images of the same objects with homogeneous, weak heterogeneous, and strong heterogeneous backgrounds (Fig. 7). Different light conditions are present in the images with non-homogeneous backgrounds.

With increasing heterogenety of the background, more erroneous correspondences are matched. If their number is very high, a false positive recognition occurs. A challenge in recognition is posed by the objects *perrier* and *truck* as they are small compared to the overall image size. With the low image resolution only few pixels remain for the object and thus only a little number of features can be extracted. During the Technical Challenge of the RoboCup we used a higher image resolution. Please refer to Sec. 3.6 for more details.

3.5 Partial Occlusion

Another experiment was performed to test the algorithm with partially occluded objects. Occlusion was simulated by partially replacing the object in the test data with the corresponding background. The results are presented in Fig. 8.

Table 2. Object recognition results on images with different backgrounds. The numbers in brackets indicate the number of false positive recognitions.

Object	hom. back.	weak het. back.	strong het. back.
bscup	100 %	90 % (1)	100 %
nizoral	100 %	100 % (2)	90 %
perrier	100 %	100 % (1)	100 % (2)
ricola	100 %	100 % (1)	100 %
truck	100 %	90 %	70 % (1)

Fig. 6. Objects from [4] used for evaluation. From left to right: *bscup*, *nizoral*, *perrier*, *ricola*, *truck*.

The unoccluded object is recognized with a total of 98 matched features and a confidence of 38 % (Eq. 11). With increasing occlusion the number of features decreases, but is still high enough for a correct recognition of the object. However, with increasing occlusion the accuracy of the computed homography (red lines in Fig. 8) and thus of the bounding box decreases.

3.6 RoboCup@Home: Technical Challenge

This object recognition approach was also applied during the Technical Challenge in the @Home league of the RoboCup world championship that took place in Mexico-City in 2012. 50 objects were placed on a table containing randomly selected 15 of 25 previously known objects. Our robot could correctly identify 12 of the 15 present known objects correctly, while at the same time having no false positive recognitions. This recognition result was achieved with a single scene view. With this result our robot places first in the Technical Challenge and won

Fig. 7. Object *nizoral* from [4] with different backgrounds. From left to right: homogeneous, weak heterogeneous, and strong heterogeneous backgrounds.

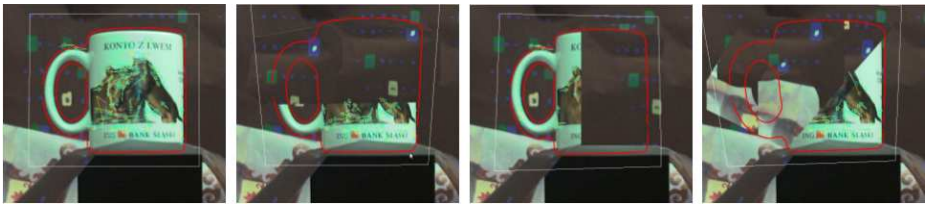

Fig. 8. Example images for detection of partially occluded objects. The unoccluded object (top left) is recognized with 98 matched features and 38 % confidence. The occluded object have less features, but are still recognized correctly: 49 with 33 % (top right), 17 with 17 % (bottom left), and 34 with 34 % (bottom right).

the Technical Challenge Award. The input image for object recognition as well as the recognition results are shown in Fig. 9.

We use a difference of 30° between object views to minimize training time and the number of images in the database. Objects were trained and recognized with an off-the-shelf digital camera (Canon PowerShot SX100 IS) and an image resolution of 8 megapixels (MP). Since the object recognition took a long processing time, further tests with the RoboCup@Home objects were performed after the Technical Challenge (Tab. 3.6). The total recognition time depends on the resolution in the training phase as well as on the resolution of the scene image. However, the resolution in the training has a greater influence on the total recognition time. According to Tab. 3.6 it is sufficient to create an object database where features are extracted from 4 MP images, but use a resolution of 8 MP for recognition. This is not surprising since the object to camera distance is usually smaller during training than during recognition. Thus, even with a lower image resolution a sufficent number of features can be extracted and saved in the database during training.

Table 3. Comparison of different image resolutions and their effect on recognition time and recognition quality.

Resolution Training [MP]	Resolution Scene [MP]	Recognition Time [s]	True Positives	False Positives
4	4	20	5	1
4	8	26	12	0
8	4	53	6	1
8	8	117	12	0

4 Conclusion

We presented our object recognition approach that we use in the RoboCup@Home league. Our recognition approach performs well on images with cluttered back-

Fig. 9. The input image for object recognition as acquired by our robot during the Technical Challenge of the RoboCup (top). Object recognition results with 12 correctly identified objects (bottom). During training and recognition an image resolution of 8 MP was used.

ground and partially occluded objects. Objects at different scales and with arbitrary poses in the scene image are recognized reliably. For best results, it is recommended to use high resolution images in order to extract sufficient features for object representation. Our future work will concentrate on further evaluating and optimizing our approach.

References

1. Bay, H., Tuytelaars, T., Gool, L.J.V.: SURF: Speeded up robust features. ECCV pp. 404–417 (2006)
2. Decker, P., Thierfelder, S., Paulus, D., Grzegorzek, M.: Dense Statistic Versus Sparse Feature Based Approach for 3D Object Recognition. Pattern Recognition and Image Analysis 21(2), 238–241 (2011)
3. Grimson, W.E.L., Huttenlocher, D.P.: On the sensitivity of the hough transform for object recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-12(3), 255–274 (1990)
4. Grzegorzek, M., Niemann, H.: Statistical object recognition including color modeling. In: 2nd International Conference on Image Analysis and Recognition. pp. 481–489. Springer, Toronto, Canada (2005)
5. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
6. Muja, M.: Flann, fast library for approximate nearest neighbors (2009), http://mloss.org/software/view/143/