Real-Time Object Recognition in Digital Images for Industrial Applications

Markus Ulrich[1,2], Carsten Steger[2], Albert Baumgartner[1], and Heinrich Ebner[1]
[1]Lehrstuhl für Photogrammetrie und Fernerkundung, Technische Universität München,
Arcisstr. 21, 80290 München
[2]MVTec Software GmbH, Neherstr. 1, 81675 München

Abstract. This paper describes a technique for real-time object recognition in digital images. On the one hand, our approach combines robustness against occlusions, distortions, and noise with invariance under rigid motion, i.e., translation and rotation, and local illumination changes. On the other hand, the computational effort is small in order to fulfil requirements of real-time applications.

A shape-based matching approach using the principle of the generalized Hough transform is employed to receive a coarse solution for position and orientation. A novel efficient limitation of the search space in combination with a hierarchical search strategy is implemented to reduce the computational effort. The quantization problems occurring in the generalized Hough transform as well as in our modifications are analyzed and their solutions are presented. To meet the demands for high precision in industrial tasks, a subsequent refinement adjusts the final parameters of position and orientation. An example and experimental results complete this paper.

Key words: Real-time, Object recognition, Generalized Hough Transform

1 Introduction

In many industrial applications, e.g., quality control, inspection tasks, or robotics, there is a particularly high demand on the object recognition approach to find the object in the image under certain aggravating circumstances. The first condition to consider is that the recognition approach must fulfil real-time requirements, i.e., the recognition task must not delay the process it is embedded in. Furthermore, the method should be highly robust against occlusion and clutter. Illumination changes often cannot be avoided completely across the entire field of view. Thus, the recognition method should additionally be robust against non-linear contrast changes. Since in a great number of industrial applications the appearance of the object to be found has limited degrees of freedom in this study only rigid motion, i.e., translation and rotation, is considered, which is sufficient in many cases.

In the literature different object recognition approaches can be found. All recognition methods have in common that they require some form of representation of the object to be found, which will be called *model* below. The data that is utilized to build the model can, for example, be extracted from a CAD representation or from one or more images of the object. In the case of a single image of the object serving as input data - as it is in our approach - this image will be refered to as *reference image*. The image in which the object should be recognized will be called *search image*. Almost all object recognition approaches can be split into two successive phases: the *offline phase* including the generation of the model and the *online phase*, in which the constructed model is used to find the object in the search image. Thus, only the computation time of the online phase is critical considering the real-time requirement.

One possibility to group object recognition methods is to distinguish between gray value based, e.g., Gonzalez and Woods (1992), Brown (1992), Lai and Fang (1999) and feature based matching approaches, e.g., Borgefors (1988), Rucklidge (1997), Huttenlocher et al. (1993) or Olson and Huttenlocher (1997). Gray value based matching has several disadvantages and does not meet most of the above mentioned demands. It is too computationally expensive for real-time applications and is not robust to occlusions or to clutter. Features, e.g., points, edges, polygons, or regions characterize the object in a more compressed and efficient way than the gray value information and thus are better suited for real-time recognition.

In our approach edges and their orientations, i.e., the shape of the object, are used as features. The basic principle of the generalized Hough transform (GHT) (Ballard, 1981) is applied, which is an efficient method to compare the class of features used in this work and therefore allows a rapid computation (see section 2). After emphasizing the major drawbacks of the GHT in section 2 we further optimize the GHT by considering modifications to fulfil industrial demands (see section 3). The quantization problems occurring in the generalized Hough transform as well as in our modifications are analyzed and their solutions are presented (see section 4). A subsequent refinement adjusts the final parameters of position and orientation (see section 5). An example and experimental results complete this paper (see section 6).

2 The Generalized Hough Transform

2.1 Principle

Ballard (1981) generalizes the Hough transform (Hough, 1962) to detect arbitrary shapes. Ballard also takes the edge orientation into account, which makes the algorithm faster and also greatly improves its accuracy by reducing the number of false positives. To perform the offline phase of the GHT, the so-called *R*-Table must be constructed using information about the position and orientation of the edges in the reference image. Assuming that $N_{\boldsymbol{p}^r}$ is the number of edge points $\boldsymbol{p}_i^r$ ($i = 1 \dots N_{\boldsymbol{p}^r}$) in the reference image and Φ_i^r are the corresponding gradient directions, both extracted by an arbitrary edge operator. Then the *R*-table is generated by choosing a reference point $\boldsymbol{o}$, e.g., the centroid of all edge points, i.e., $x_{\boldsymbol{o}} = 1/N_{\boldsymbol{p}^r} \sum x_{\boldsymbol{p}_i^r}$, $y_{\boldsymbol{o}} = 1/N_{\boldsymbol{p}^r} \sum y_{\boldsymbol{p}_i^r}$, calculating $\boldsymbol{r}_i = \boldsymbol{o} - \boldsymbol{p}_i^r$ for all points and storing $\boldsymbol{r}$ as a function of Φ. If the orientation of the shape in the search image is not constant, i.e., the object may undergo rigid motions, for every possible orientation a separate *R*-table must be constructed.

Assuming the case of rigid motion, in the online phase a three dimensional initialized accumulator array *A* is set up over the domain of parameters, where the parameter space is quantized and range restricted. Each finite cell of this array corresponds to a certain range of positions and orientations of the model in the search image, which can be described by the three variables *x*, *y*, and θ. Here, *x* and *y* describe the translated position of $\boldsymbol{o}$ in the search image and θ the orientation of the object in the search image relative to the object in the reference image. For each edge pixel $\boldsymbol{p}_j^s$ in the search image and each *R*-table corresponding to one orientation θ_k, all cells $\boldsymbol{r}_i + \boldsymbol{p}_j^s$ in *A* receive a vote, i.e., they are incremented by 1, within the corresponding two dimensional hyper plane defined by $\theta = \theta_k$ under the condition that $\Phi_i^s = \Phi_i^r$. Maxima in *A* correspond to possible instances of the object in the search image.

2.2 Major Drawbacks

One weakness of the GHT algorithm is the - in general - huge parameter space. This requires large amounts of memory to store the accumulator array as well as high computational costs in the online phase caused by the initialization of the array, the incrementations, which approximately increase quadratically with the object size, and the search for maxima after the incrementation step. In addition, the properties of the GHT lead to the fact that the accuracies achieved for the returned parameters depend on the quantization of translation and rotation. On the other hand, in practice the quantization cannot be chosen arbitrarily finely taking again memory requirements and computation time into account.

In the following sections we tackle the above mentioned problems: A hierarchical search strategy in combination with an effective limitation of the search space is introduced. Problems concerning quantization are solved and a technique is presented to refine the returned parameters without noticeably decelerating the online phase.

3 Optimizing the Generalized Hough Transform

3.1 Hierarchical Strategy

To reduce the size of the accumulator array and to speed up the online phase both, the model and the search image are treated in a hierarchical manner. First, an image pyramid of the reference image is generated. Each pyramid level is rotated by all possible orientations of the object in the search image using $\boldsymbol{o}$ as fix point. Then, the edge amplitude and the gradient direction are computed from the rotated image using the Sobel filter. We prefer using the Sobel filter because it represents a good compromise between computational time and accuracy. Its anisotropic response and its worse accuracy can be balanced by choosing an adequate quantization of the gradient directions, which will be explained in section 4. The edge pixels are extracted by thresholding the gradient amplitude.

The number of pyramid levels N_l must be chosen in a way that in the top level the characteristic structure of the object is still recognizable and enough edge pixels are present to achieve a meaningful result.

While building the model, we have to distinguish between the top level of the reference image and the lower levels. In the online phase the recognition process starts on the top pyramid level without any a priori information about the transformation parameters x, y and θ available. The cells in A that are local maxima and exceed a certain threshold are stored and used to initialize approximate values on the lower levels. Therefore, only on the top level for each rotation one R-table is built as described in section 2.1, whereas on the lower levels a modified approach is necessary to take advantage of the a priori information returned from the next higher level.

3.2 Blurred Region

The use of a hierarchical model and the use of an image pyramid enable efficient limitations of the search space on lower pyramid levels. If the approximate values $\tilde{x}$, $\tilde{y}$, and $\tilde{\theta}$ are known, not all the edge pixels in the current pyramid level need to be extracted. To obtain an optimal search region the model shape is blurred using the uncertainties of the a priori parameters. The strategy is illustrated in Figures 1, 2, and 3. The shape is overlaid on the search image at the approximate position and orientation (Fig. 1). The a priori information is displayed as a dark gray box representing the maximum error of the approximate translation values from the level above, which will be refered to as the *approximate zone*. The positioning error $\delta x, \delta y$ is taken into account by dilating the shape with a rectangular mask of size

$(2\delta x+1)\times(2\delta y+1)$ (Fig. 2). The *blurred region* is finally obtained by successively rotating the dilated shape in both directions until the maximum amplitudes of the orientation error $\pm\delta\theta$ are reached and merging the resulting regions (Fig. 3).

The blurred regions are calculated for every quantized orientation in the offline phase and stored together with the model. In the online phase the blurred region with orientation $\tilde{\theta}$ simply must be centered at the position $\tilde{x}$, $\tilde{y}$ in the search image. Thus, the edge extraction can be restricted to the pixels lying beneath the blurred region, which greatly reduces the computational effort. In addition, the size of the accumulator array A can be narrowed to the dimensions according to the uncertainties $\delta x, \delta y$, and $\delta\theta$ of the a priori parameters, which decreases the memory amount drastically.

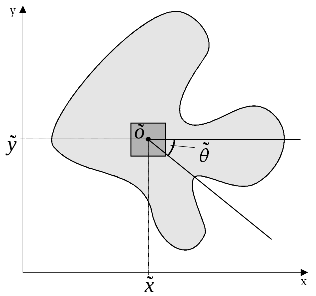

Fig. 1. Approximate values are given from the level above.

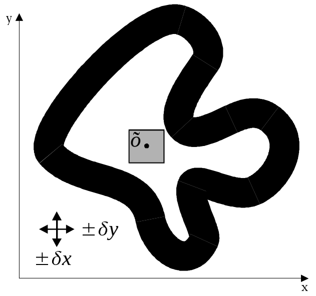

Fig. 2. Taking the translation error into account: Blurring by dilating.

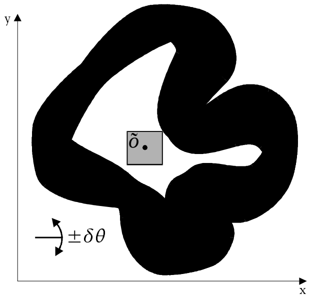

Fig. 3. Taking the orientation error into account: Blurring by rotating.

3.3 Tile Structure

After the edge extraction the second improvement is utilized. The principle of the conventional GHT is shown in Figure 4. The edge pixels $\boldsymbol{p}_1^r$, $\boldsymbol{p}_2^r$, and $\boldsymbol{p}_3^r$ have identical gradient directions. Thus, if any of those edge pixels is processed in the online phase of the conventional GHT each of the three vectors $\boldsymbol{r}_1, \boldsymbol{r}_2$, and $\boldsymbol{r}_3$ is added and the corresponding three cells are incremented.

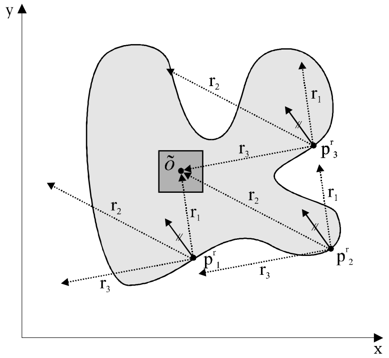

Fig. 4. The conventional GHT without using the a priori information of the translation parameters. Many unneccesary increments are executed.

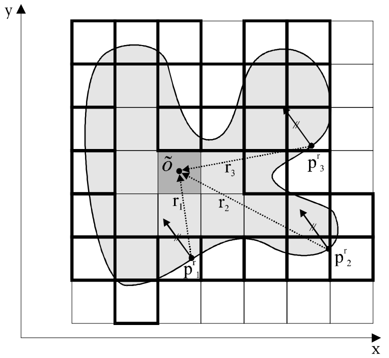

Fig. 5. Taking advantage of the a priori information using a tile structure. For each occupied tile (bold border) a separate R-table is generated.

One possible solution is to check during the voting process whether the added vectors fall in the approximate zone or not. This query and the summation of the vectors would take too much time to ensure real-time qualification. Therefore, the opposite direction is performed: Already in the offline phase the information about the position of the edge pixels relative to the centroid is calculated and stored in the model. This is done by overlaying a grid structure over the rotated reference image and splitting the image into tiles (Fig. 5). For every tile that is occupied by at least one edge pixel an *R*-table is generated. The vectors $\boldsymbol{r}_i$ are then stored in the tile in which the corresponding edge pixels fall. The consequence is that on the lower pyramid levels the model consists of a multitude of *R*-tables: For every quantized orientation and for every occupied tile a separate *R*-table is created. In the online phase the current tile is calculated using the approximate translation parameters and the position of the current edge pixel. Only the vectors in the respective tile with the appropriate gradient direction are used to calculate the incrementation cells.

4 Problems and Solutions Concerning Quantization

When applying the principle of the GHT several problems occur concerning the quantization of the parameters and the gradient directions. A similar difficulty occurs using the tile structure described in section 3.3. In the following section we analyze these problems and present our solutions.

4.1 Rotation

The number of discerned discrete orientations considered when building the model depends on the shape of the object especially on the distance between the edge pixels and the centroid. In general, the step size $\Delta\theta$ must be chosen the smaller the bigger the searched object is. To ensure that all vectors $\boldsymbol{r}_i$ of the model hit the same cell of the parameter space in the online phase the maximum positioning error ε in *x* and *y* must not exceed ½ pixel:

$$\Delta\theta = \frac{2\varepsilon}{r^{\max}}$$

Here, $r^{\max}$ is the maximum of the distances between all edge points and the centroid. Hence, large objects would lead to a very fine quantization resulting in large memory amounts and time consuming computations. Therefore, and because $r^{\max}$ is not representative for the whole model, a more sophisticated computation is applied, which takes all edge pixels into account: If $\Delta\theta$ is chosen too large, the peak in the corresponding cell in *A* will be weakened because some of the vectors $\boldsymbol{r}_i$ will miss the cell. The degree of weakness η is computed as

$$\eta(\Delta\theta) = \frac{\left|\left\{\boldsymbol{r} \mid r > \frac{2\varepsilon}{\Delta\theta}\right\}\right|}{N_{p^r}},$$

where $\varepsilon = 0.5$. The maximum possible peak height $\Gamma^{\max}$ will then be reduced to $\Gamma(\Delta\theta) = \eta(\Delta\theta) \cdot \Gamma^{\max}$. The computational effort Ω increases linearly with the number of discrete orientations $2\pi / \Delta\theta$. Therefore, it also can be written as function of $\Delta\theta$:

$$\Omega(\Delta\theta) = \frac{1}{\Delta\theta}.$$

To find the optimal value for $\Delta\theta$ we have to minimize the computational effort while maximizing the peak height $\Gamma(\Delta\theta)$. This is obtained by maximizing the ratio $\Gamma(\Delta\theta)/\Omega(\Delta\theta)$.

4.2 Translation

The height of the peak in *A* depends on subpixel translations of the object. Under ideal conditions the peak in the parameter space is equal to N_{p^r}. If the object in the search image is translated by subpixel values in *x* and *y* direction relative to its position in the reference image the peak height decreases because the votes are distributed over more than one cell. This effect reaches its maximum at a subpixel translation of ½ pixel in each direction, which reduces the peak to 25 percent of its original height. The left part of Figure 6 illustrates this behavior. Under the assumption that the neighborhood of the peak is rotational symmetric the peak height can be made independent of subpixel translation by smoothing the translation hyper planes, i.e., $\theta = \text{const.}$, of the parameter space applying a mean filter of size 2×2 (see right part of Figure 6).

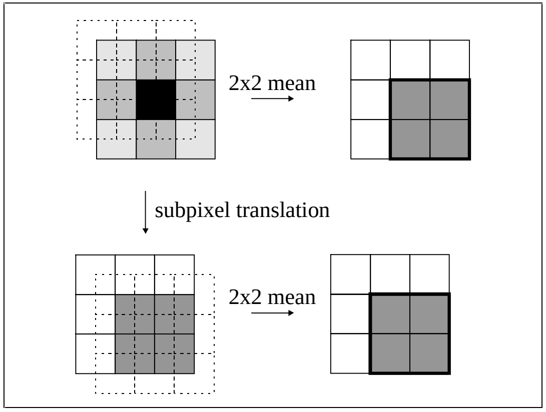

Fig. 6. Subpixel translation decreases the peak height (left). This effect is eliminated by smoothing, applying a 2×2 mean filter (right).

4.3 Gradient Direction

The right quantization of the gradient direction intervals within the *R*-tables depends on various factors. The determination of the interval defines the range of gradient directions that are treated as equal. The smaller the interval the faster the computation. On the other hand, an interval that is chosen too small leads to instable results. The first thing to consider is the variance of the gradient direction due to noise in the image. The gradient directions are computed out of the partial directional derivatives returned by the Sobel filter. Let σ_{gray} be the standard deviation of the gray values in the search image and *T* the minimum edge amplitude, i.e., the threshold for the edge amplitudes in the edge extraction step. Then, the standard deviation of the gradient directions σ_{grad} is calculated as

$$\sigma_{\text{grad}}[\text{rad}] = \frac{\sqrt{3}}{2} \cdot \frac{\sigma_{\text{gray}}}{T},$$

using the law of error propagation. σ_{gray} depends on the utilized camera and can be determined experimentally. A typical range for σ_{gray} is [1.5,2]. Assuming a normal distribution the minimal gradient interval can be determined by specifying the desired percentage of gradient directions lying in the interval. For example, if 95 percent should fall inside the interval then its boundaries are $[-2\sigma_{\text{grad}}, +2\sigma_{\text{grad}}]$.

The second influence that must be taken into account is the inherent absolute accuracy of the Sobel filter, i.e., the difference between the real partial derivatives and the Sobel response. Since the Sobel filter is an anisotropic filter its absolute accuracy depends on the current gradient direction. Experimental results have shown that the maximum error that affects the gradient direction occurs at the expected angles of $n \cdot (\pi / 8)$, where $n = 2z + 1$ and $z \in \mathbb{Z}$ reaching amplitudes of about ± 4 degrees. Thus, combining the two described effects the interval can be chosen as $[-\Delta\Phi / 2, +\Delta\Phi / 2]$, where $\Delta\Phi = 2\max\{4° \cdot (\pi / 180°), 2\sigma_{\text{grad}}\}$.

Another factor, which affects the gradient direction is subpixel translation. Taking the edges by 1D curves in the image, the magnitude of the gradient variation directly depends on the second derivative of the edge curve, i.e., the curvature of the edges. In Figure 7 an example of subpixel translation in *y* direction shows this effect. Here, the gradients of the corner pixel and the pixel below are modified. One possible solution for this problem is to introduce only stable edge points into the model, whose gradient direction at most vary in a small range. The stable points can be found by translating the reference image by ½ pixel in each direction and comparing the computed gradient direction Φ^t with those of the untranslated image Φ^0. The edge pixels with small differences, i.e., $\delta\Phi = \left|\Phi^0 - \Phi^t\right| < \delta\Phi^{\max}$, form the model. $\delta\Phi^{\max}$ should be chosen as $\delta\Phi^{\max} = \Delta\Phi/2$ to ensure that no gradient direction is missing its interval.

A final important detail is how to avoid boundary effects of the gradient intervals. Already due to small rotations of the object in the search image the gradient directions may cross the boundary to the next interval and therefore are not considered in the voting process. This would lead to a drastical decrease of the corresponding peak in the accumulator array. In order to avoid this behavior the gradient intervals must overlap each other. In practice, the overlapping can be achieved by copying the gradient vectors to the nearest adjacent interval. Consequently every vector in the *R*-table must be stored twice, which leads to an increasing computational effort in the online phase. However, there is no other way to avoid this boundary effect.

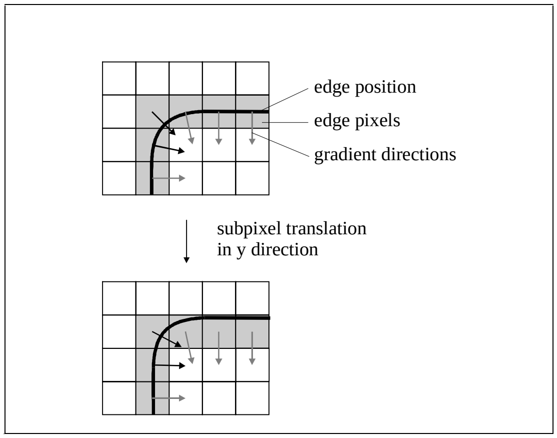

Fig. 7. Subpixel translation leads to variations in the gradient directions, particularly in regions with high edge curvature.

4.4 Tile Structure

A problem similar to the quantization of the gradient directions occurs when using the tile structure described in section 3.3. The size of the tiles should be chosen such that the uncertainty of the approximate position is taken into account, i.e., the dimension of the tiles in the *x* and *y* direction should be $2\delta x$ and $2\delta y$. Furthermore, it must be ensured that an error of $2\delta x$ and $2\delta y$ of the approximate position $\tilde{\boldsymbol{o}}$ does not result in omitting the relevant edge pixels as a consequence of considering the wrong tile. This problem is shown in Figure 8.

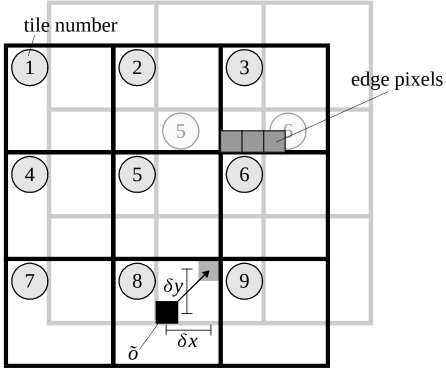

Fig. 8. The edge pixels of the search image are contained in tile 3. Taking the uncertainty of the approximate position into account, the containing tile can change. Here, now the tiles 5 and 6 are occupied.

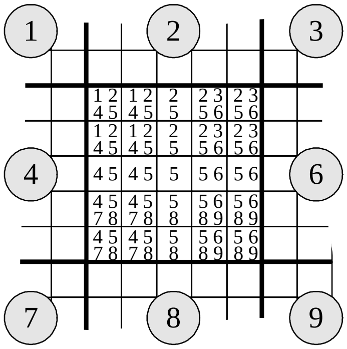

Fig. 9. The neighboring tiles to be checked are stored for every pixel in an index array within the tile structure. This enables a fast access in the online phase.

In this example the three edge pixels are stored in tile 3 within the model. However, in the online phase the possibly incorrect approximate position of $\tilde{\boldsymbol{o}}$ leads to the fact that the calculated tiles would be 5 and 6. This would result in omitting the three edge pixels in the voting process. The solution is illustrated in Figure 9. To avoid omitting relevant tiles certain neighboring tiles must be considered additionally. Since not all of the neighbors need to be involved to ensure the above mentioned requirement an index array is constructed holding the relevant tiles to be checked. For each pyramid level within the model one index array is computed in the offline phase. For each edge pixel in the search image the corresponding tiles are investigated by looking up the index array. This facilitates a fast computation while keeping the memory requirement low. Here, the method of making multiple entries like in the case of the gradient direction intervals would result in a almost four-fold memory requirement.

5 Refinement of Position and Orientation

The accuracy of the results of the GHT on the lowest pyramid level depends on the chosen quantization of the parameter space. To refine the parameters of position and orientation we use the principle of the 3D facet model (Haralick and Shapiro, 1992). The 3D parameter space *A* is assumed to be a 3D piecewise continuous intensity surface $f(x', y', \theta')$, in which the intensities are represented by the number of entries in the cells of the accumulator array. The refinement of the parameters can be done by extrapolating the maximum of the continuous function in the neighborhood of the maximum of *A*.

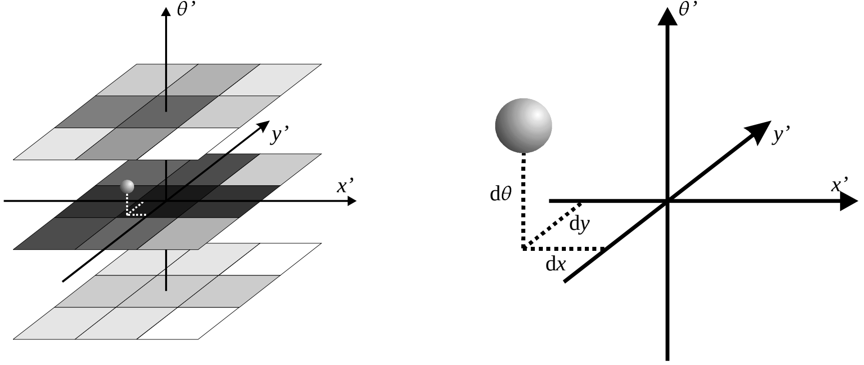

Fig. 10. Applying a 3D facet model on the 3×3×3 neighborhood of the maximum in the discrete parameter space *A* (left hand). The corrections dx, dy, $d\theta$ are added to the discrete values of the maximum (right hand).

We choose a local second order polynomial

$$f(x', y', \theta') = k_1 x' + k_2 y' + k_3 \theta' + k_4 x'^2 + k_5 x' y' + k_6 x' \theta' + k_7 y'^2 + k_8 y' \theta' + k_9 \theta'^2$$

to fit the discrete values. The reason for the choice of a second order polynomial is the existence of an extremum we are interested in. The coefficients $k_1 \dots k_9$ are calculated by convolving *A* with the appropriate 3D model masks (Steger, 1998), which can be computed using a least-squares fit, at the position of its local maximum (*x*, *y*, and θ). The cells in a $3 \times 3 \times 3$ neighborhood of the local maximum in *A* serve as sampling points for the calculation. The principle is shown in Figure 10. After fitting the polynomial the coordinates of its maximum *dx*, *dy*, $d\theta$ are calculated and used to correct the discrete values. This refinement does not carry weight considering the computational time because the only thing to do is solving a linear equation system, which can be described by a 3×3 matrix.

6 Experimental Results

To validate the accuracy of the resulting parameters *x*, *y*, and θ we generated some image sequences containing subpixel translations and rotations of an object. The reference image of size 652×494 is shown in Figure 11. In the Figures 12 and 13 the user-selected object of size

240×130 pixels and its extracted shape are illustrated. Figure 14 shows an example search image of the rotated object and the result of our recognition approach.

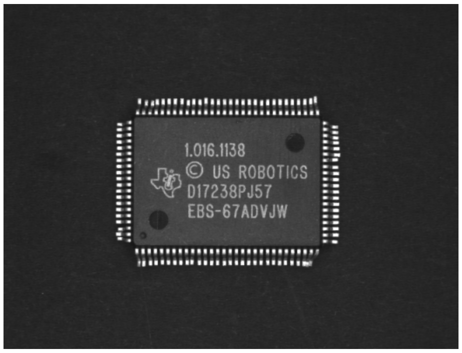

Fig. 11. Reference Image.

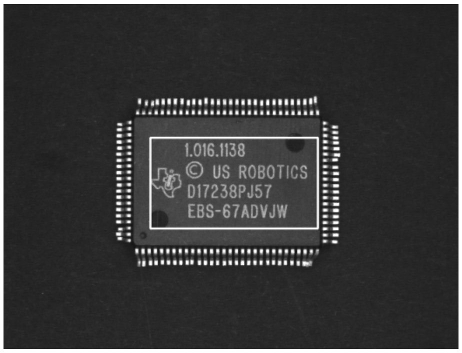

Fig. 12. User defined object.

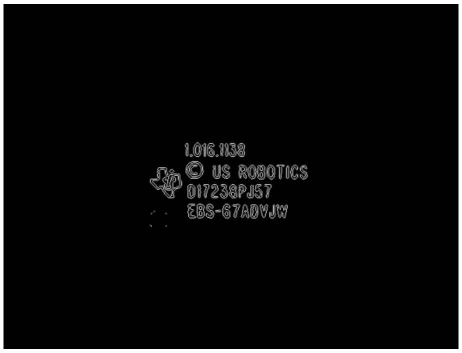

Fig. 13. Extracted object shape.

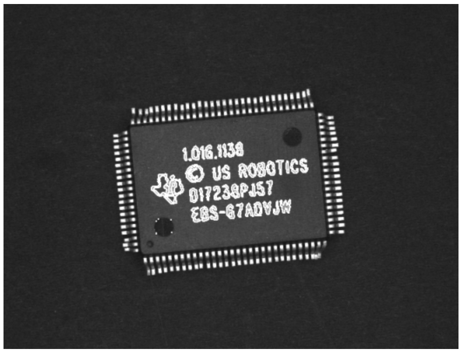

Fig. 14. Search image and located object instance.

Figures 15, 16, and 17 show the errors of position and orientation in combination with their standard deviations for three image sequences, which contain subpixel translation and rotation. Our approach is able to locate objects with an average error of about 0.03 pixels in position and 0.05 degrees in orientation. The maximum errors are approximately 0.1 pixels, and 0.12 degrees. After adding white noise with a maximum amplitude of ±5 to the search image these values degraded only slightly. Furthermore the approach is robust considering that occlusions merely decrease the peak in the accumulator array proportional to the percentage of occlusion. To show the real-time capability: the average computational time needed to find the object, which contains approximately 3000 model points in the lowest pyramid level, is about 60 msec on a PENTIUM III with 667 MHz.

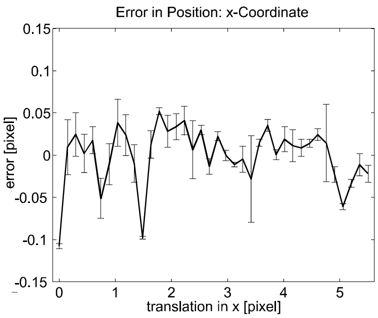

Fig. 15. Subpixel translation in x direction.

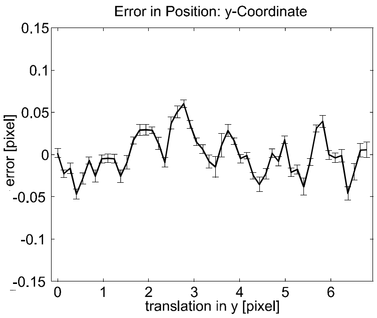

Fig. 16. Subpixel translation in y direction.

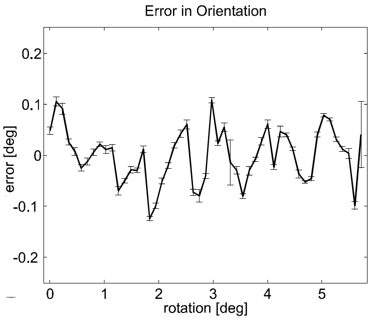

Fig. 17. Rotation.

7 Summary

By using a hierarchical search strategy in combination with a new effective search space limitation our approach fulfils the requirements of real-time. Since the object's shape does not depend on the illumination, this method in addition is robust against illumination changes to a certain extent. Furthermore, it is also extremely robust against partial occlusion and clutter, as the raw gray value information is not used directly. The coarse solution of the position and orientation parameters of the object in the search image is adjusted in a subsequent refinement to meet the demands for high precision and accuracy in industrial applications. This approach satisfies the requirements and is also general with regard to the type of objects that can be recognized: a representation (model) of the object is automatically generated solely from one image of the object itself. No further information needs to be added to the model manually by the operator

References

1. Ballard, D. H.. Generalizing the Hough transform to detect arbitrary shapes. Pattern Recognition, 1981, 13(2): 111-122

2. Borgefors, G.. Hierarchical chamfer matching: A parametric edge matching algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1988, 10(6): 849-865

3. Brown, L. G.. A survey of image registration techniques. ACM Computing Surveys, 1992, 24(4): 325-376

4. Gonzalez, R. C., Richard, E. W.. Digital Image Processing. Addison-Wesley Publishing Company, 1992: 583-586

5. Haralick, R. M., Shapiro, L. G.. Computer and Robot Vision. Volume 1. Addison-Wesley Publishing Company, 1992. 371-452

6. Hough, P. V. C.. Method and means for recognizing complex patterns. U.S. Patent 3,069,654, 1962

7. Huttenlocher, D. P., Klanderman, G. A., Rucklidge, W. J.. Comparing Images Using the Hausdorff Distance. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1993, 15(9): 850-863

8. Lai, S.,Fang, M.. Accurate and fast pattern localization algorithm for automated visual inspection. Real-Time Imaging, 1999, 5: 3-14

9. Olson, C. F., Huttenlocher, D. P.. Recognition by Matching With Edge Location and Orientation. Automatic target recognition by matching oriented edge pixels. IEEE Transactions on Image Processing, 1997, 6(1): 103-113

10. Rucklidge, W. J.. Efficiently locating objects using the Hausdorff distance. International Journal of Computer Vision, 1997, 24(3): 251-270

11. Steger, C.. Unbiased Extraction of Curvilinear Structures from 2D and 3D Images. Fakultät für Informatik, Technische Universität München, Dissertation, Herbert Utz Verlag, 1998. 92-96