Detection of Low-textured Objects

Christopher Bulla and Andreas Weissenburger
Institut für Nachrichtentechnik, RWTH Aachen University, Aachen, Germany

Keywords: Object Detection, Local Features, SIFT.

Abstract: In this paper, we present a descriptor architecture, SIFText, that combines texture, shape and color information in one descriptor. The respective descriptor parts are weighted according to the underlying image content, thus we are able to detect and locate low-textured objects in images without performance losses for textured objects. We furthermore present a matching strategy beside the frequently used nearest neighbor matching that has been especially designed for the proposed descriptor. Experiments on synthetically generated images show the improvement of our descriptor in comparison to the standard SIFT descriptor. We show that we are able to detect more features in non-textured regions, which facilitates an accurate detection of non-textured objects. We further show that the performance of our descriptor is comparable to the performance of the SIFT descriptor for textured objects.

1 INTRODUCTION

A problem often occurring in computer vision tasks is the detection or recognition of identical objects within different images. In recent years, invariant local features (Lowe, D.G., 2004; Bay et al., 2006) have been introduced to solve this problem. A local feature is a n-dimensional vector, that contains a unique description of an image patch. An image typically contains several local features and the entirety of all local image features can be regarded as the description of the image. If two or more images contain identical objects a subset of their features will be identical or similar. This could be used to detect common objects across different images (Bulla and Hosten, 2013).

Many local feature approaches (e.g. SIFT, SURF) are designed to detect meaningful image patches and describe their gradient distribution. Therefore, they could be well used to describe textured objects or images (cf. figure 1(a)). When they are used to describe low-textured objects (cf. figure 1(b)) two problems occur: Firstly, low-textured objects only have high gradients at their border. Therefore most of the local image patches are detected at the border and not on the object itself. Secondly, when the patches at the border are described, a large amount of the information covered by the feature characterizes the background. Since low-textured objects usually have a unique gradient-less surface, the part of the feature that covers the object doesn't contribute much to the description.

In the literature several approaches have been introduced to describe low-textured objects. The *Bunch Of Lines Descriptor* (Tombari et al., 2013) for example approximate the contour of an object by polygons and describes this polygons. The *Boundary Structure Segmentation* approach (Toshev et al., 2012) uses the *chordiogram*, a multidimensional histogram, to describe a by superpixel segmented image and the *Distinct Multi-Colored Region Descriptors* approach (Naik and Murthy, 2007) describes the objects by regions with different colors. All approaches are indeed able to describe low-textured objects, but have in common that they lower the ability to describe textured objects.

In this paper we present a combination of three descriptors in order to describe low-textured, as well as textured objects. We thereby combine texture, shape and color information into a single descriptor and weight each descriptor component based on the image content. In textured regions e.g. we give a higher weight to the texture descriptor, while in non-textured regions the shape descriptor gets a higher weight. We furthermore use a matching strategy apart the well known nearest neighbor matching which allows an accurate object localization.

The rest of the paper is organized as follows. In chapter 2 we present our descriptor architecture. The matching strategy will be presented in chapter 3. These are evaluated in chapter 4. Finally conclusions will be drawn in chapter 5.

Bulla C. and Weissenburger A..
Detection of Low-textured Objects.
DOI: 10.5220/0005269602650273
In *Proceedings of the 10th International Conference on Computer Vision Theory and Applications* (VISAPP-2015), pages 265-273
ISBN: 978-989-758-090-1

2 DESCRIPTOR ARCHITECTURE

Our descriptor, SIFText has been designed to increase the performance of SIFT in low-textured regions, while maintaining its performance in textured regions. We therefore combine texture, shape and color information into a single descriptor and weight each descriptor part by a locally calculated value, derived from the texture of the image. As texture descriptor we use SIFT, the shape will be described with ART coefficients and the color descriptor is a combination of Hue- and Opponent-Angle histograms. The single descriptor parts as well as their weighting will be described in the following subsections.

2.1 SIFT

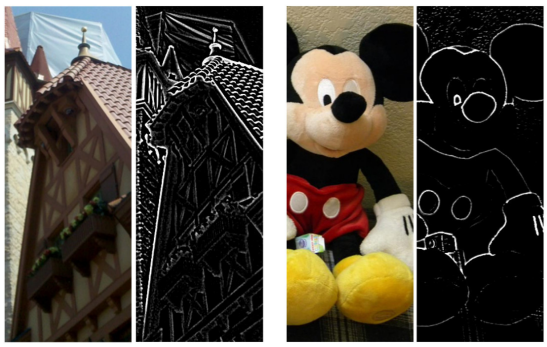
Scale Invariant Feature Transform (SIFT) has been published in 2004 by Lowe (Lowe, D.G., 2004) and describes the content of an image patch by a histogram of gradient orientations.

The selection of feature points is done in a scale invariant way. In a first step the image $I(x,y)$ is convolved with variable scale Gaussians $G(x,y,\sigma)$, and the Difference-of-Gaussians (DoG) $D(x,y,\sigma)$ is calculated:

$$D(x,y,\sigma) = I(x,y) * [G(x,y,k\sigma) - G(x,y,\sigma)] = L(x,y,k\sigma) - L(x,y,\sigma)\ . \quad (1)$$

Possible feature positions are given by the extrema of these DoG images across all scales and image resolutions. A refinement step selects the most distinctive feature points and interpolates their exact position.

To make the descriptor invariant to object orientation, one or more dominant orientations are calculated for each feature point. The dominant orientations are given by the peaks of a weighted orientation histogram,



(a) Textured image and its gradients (b) Low-textured image and its gradients

Figure 1: Gradients of a textured and a low-textured image.

which is formed from the gradient orientations $\Theta(x,y)$ of sample points within a region around the point:

$$\Theta(x,y) = \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right). \quad (2)$$

The weighting factors are given by the gradient magnitude $m(x,y)$:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2)} + \sqrt{(L(x,y+1) - L(x,y-1))^2}\ . \quad (3)$$

To build the descriptor the region around the feature is divided into several sub-region and again, a weighted histogram of gradient orientations is calculated for each sub-region. The descriptor is given by the concatenation of all sub-histograms.

2.2 Shape

Low-textured objects can be well described by their shape. We use *Maximally Stable External Regions* (MSER) (Matas et al., 2004) to detect significant shapes in an image and describe them with a set of *Angular Radial Transform* (ART) (Bober, 2001) coefficients.

Maximally Stable Extremal Regions are connected regions that are characterized by an approximately uniform intensity surrounded by a contrasting background. Furthermore, they are stable across a range of thresholds on the intensity function. In order to avoid segmentation errors on the edges of the MSER regions (e.g. due to blurring artifacts in case of large scale changes), we detect the MSER regions, similar to SIFT, on multiple octaves (cf. (Forssen and Lowe, D.G., 2007)). A morphological closing filter is applied on the detected region, subsequent to the detection to minimize noise within the region.

Figure 3 shows the detected MSER regions for one image and visualizes the difference between a MSER region and and a feature that has been detected on the textured area of the image.

To calculate a descriptor from the maximal stable regions we use ART, which describes the shape of a region with the coefficients $F_{m,n}$:

$$F_{n,m} = \int_0^{2\pi}\int_0^1 V^*_{n,m}(\rho,\Theta) I(\rho,\Theta)\rho\ \delta\rho\ \delta\Theta \quad (4)$$

In equation 4, $I(\rho,\Theta)$ is an image in polar coordinates (e.g. a MSER region) and $V^*_{n,m}$ a ART basis function:

$$V_{n,m}(\rho,\Theta) = \frac{1}{2\pi} e^{jm\Theta} R_n(\rho) \quad (5)$$

$$R_n(\rho) = \begin{cases} 1, & \text{if } n = 0 \\ 2\cos(\pi n\rho), & \text{if } n \neq 0 \end{cases} \quad (6)$$

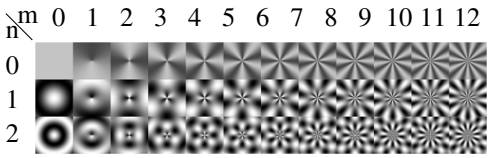


Figure 2: ART basis functions (real part).

Figure 2 depicts exemplarily the real parts of the fist 36 ART basis coefficients.

The coefficients are scale and translation invariant. A rotation causes a shift in the coefficients phase. To achieve rotation invariance we only use the coefficients real part.

2.3 Color

The combination of SIFT and color has been extensively studied in the literature (Van De Weijer and Schmid, C., 2006). We used a combination of Hue- and Opponent-Angle histograms in our descriptor architecture. The Hue descriptor is suited for the description of high-saturated images, while the Opponent-Angle descriptor is proper for the description of low-saturated images.

Hue is the angle between the difference of RGB color channels and can be computed as:

$$Hue = \text{atan}\left(\frac{\sqrt{3}(G-B)}{2R-G-B}\right) . \tag{7}$$

The Opponent-Angles are the partial derivatives in x- and y- direction of the Opponent-Colors:

$$O_1 = \frac{1}{\sqrt{2}(R-G)}, \; O_2 = \frac{1}{\sqrt{6}}(R+G-2B) \tag{8}$$

2.4 Descriptor Weighting and Fusion

In order to combine the individual descriptors, it is necessary to normalize them. Therefore, each descriptor will be normalized to unit length and the final descriptor is a weighted combination of the three normalized descriptors. The weighting factors steer the influence of each individual descriptor to the total descriptor. We choose the weighting factor for each feature point adaptively, based on the image content. In the left part of figure 3 a feature on a textured region is depicted. Here, we have distinctive gradient information, but no maximal stable region. Therefore the SIFT descriptor for this feature point should have more influence on the total descriptor than the shape descriptor. The opposite holds for the feature point in the right part of figure 3.

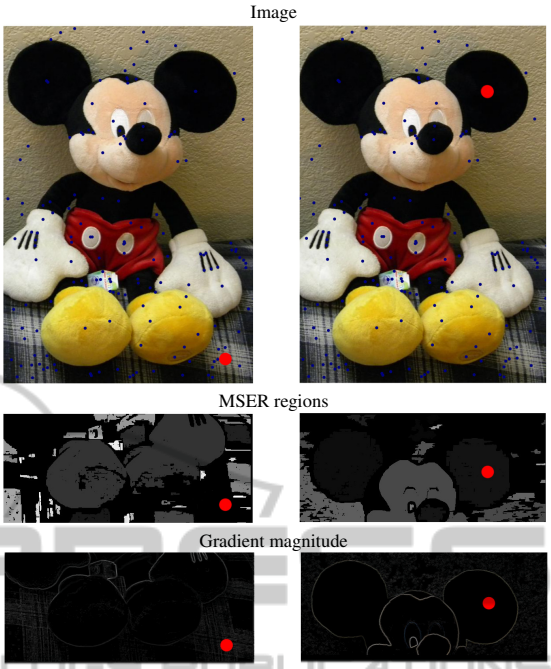


Figure 3: Difference between MSER regions and gradient based regions. Left: Keypoint in textured area with almost no shape information. Right: MSER region with almost no gradient information.

2.4.1 Weighting Factors for SIFT and SHAPE

The weighting factors for textured and non-textured regions will be defined through the *Gray Level Co-occurrence Matrix* (GLCM). The GLCM $\mathbf{C}^{\delta}_{i,j}$ is the distribution of co-occurring intensity values (i, j) at a given offset $\delta = (\delta_x, \delta_y)$ in an image of size (I_m, I_n):

$$\mathbf{C}^{\delta}_{i,j} = \sum_{p=1}^{I_n}\sum_{q=1}^{I_m} \begin{cases} 1, \text{ if } i(p,q) = i \wedge I(p+\delta_x, q+\delta_y) = j \\ 0, \text{ else.} \end{cases} \tag{9}$$

The weighting factor α can, with the number of gray levels N_G be computed as:

$$\alpha = \frac{\sum_{i=0}^{N_G}(\mathbf{C}^{1,1}_{i,i})^2}{\sum_{i=0}^{N_G}\sum_{j=0}^{N_G}(\mathbf{C}^{1,1}_{i,j})^2} . \tag{10}$$

For a homogeneous region, α will be equal to one, while for a checkerboard pattern α will converge to zero. The behaviour of alpha for some synthetically generated texture pattern is depicted in figure 4. To precisely steer the impact of α on the individual descriptor parts, we mapp α to a sigmoid. As result of this mapping we get the weighting factor a:

$$a = \frac{1}{1+e^{-b\alpha-\sigma_0}} . \tag{11}$$

The parameter b in equation 11 could be used to create either linear or threshold-like behavior of the sigmoid,

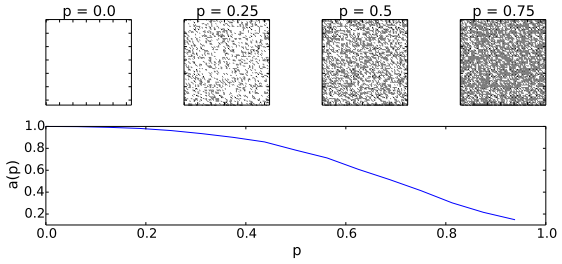


Figure 4: GLCM example.

while σ_0 could be used to prefer one descriptor part. The normalized shape and SIFT descriptors $\mathbf{f}'^{SHAPE}$ and $\mathbf{f}'^{SIFT}$ are thus given by:

$$\mathbf{f}'^{SHAPE} = a\frac{\mathbf{f}^{SHAPE}}{\sum_{j=0}^{N_{SHAPE}}|\mathbf{f}_j^{SHAPE}|} \quad (12)$$

$$\mathbf{f}'^{SIFT} = (1-a)\frac{\mathbf{f}^{SIFT}}{\sum_{j=0}^{N_{SIFT}}|\mathbf{f}_j^{SIFT}|}, \quad (13)$$

with N_{SHAPE}, N_{SIFT} denoting the number of elements in the shape and texture descriptor

2.4.2 Weighting Factor for COLOR Descriptor

The color descriptor itself has, due to its construction, an internal weighting, which reacts adaptively on the saturation of the image region. Nevertheless, for features where the shape as well as the texture descriptor cover significant information (e.g. for $a = 0.5$), we lower the influence of the color descriptor with a weighting factor c which is a function of a:

$$c(a) = \frac{1}{1+e^{-|b\alpha+\sigma_0|}}. \quad (14)$$

The run of $c(a)$ for different values of b is depicted in figure 5.

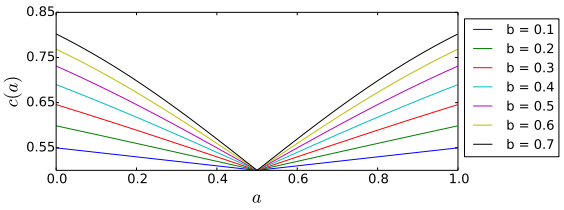
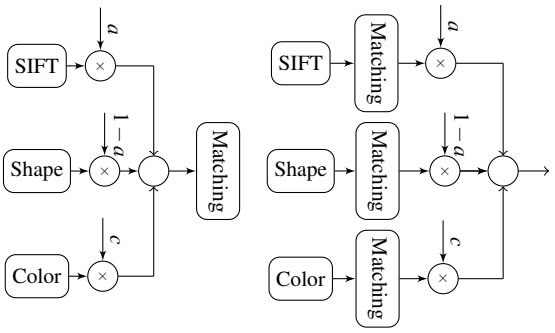


Figure 5: Color descriptor weighting factor $c(a)$, for different values of b.

2.4.3 Descriptor Fusion

The combination of different types of descriptors could be done either before (*early fusion*) or after (*late fusion*) the matching stage. Depending to the fusion method, the descriptor weighting has to be done before



(a) Early descriptor fusion (b) Late descriptor fusion

Figure 6: Descriptor fusion.

or after the matching. Figure 6 depicts schematically the early and late fusion. Since the quantisation noise in the early fusion is in general higher that in the late fusion, we decided to use the latter. The distance d between two feature descriptors $\mathbf{f}^q$ and $\mathbf{f}^t$ is thus given as:

$$\begin{aligned} d = & \sqrt{a\sum_{i=0}^{N_{SIFT}}(\mathbf{f}_i^{q,SIFT}-\mathbf{f}_i^{t,SIFT})^2} \\ + & \sqrt{(1-a)\sum_{j=0}^{N_{SHAPE}}(\mathbf{f}_j^{q,SHAPE}-\mathbf{f}_j^{t,SHAPE})^2} \\ + & \sqrt{c\sum_{k=0}^{N_{COLOR}}(\mathbf{f}_k^{q,COLOR}-\mathbf{f}_k^{t,COLOR})^2}, \end{aligned} \quad (15)$$

with N_{SIFT}, N_{SHAPE} and N_{COLOR} denoting the length of the descriptors.

1

3 DESCRIPTOR MATCHING

Usually, feature based object detection algorithms perform a nearest neighbor matching followed by an outlier rejection algorithm (e.g. *RANdom SAmple Consensus* (RANSAC) (Fischler and Bolles, 1981)) to determine corresponding features. The outlier rejection assumes that all correct correspondences conform to an affine or perspective transformation $\mathbf{T}$. While this assumption is useful for images that dont't have repetitive structures, it is not practical for images with low-textured objects, which often have repetitive or similar structures (cf. figure 3). These similar structures could lead to correspondences that will be removed in the outlier rejection stage, because they don't conform to the calculated transformation even if they are correct. Similar structures could furthermore lead to ambiguities in the matching process.[1]

[1]For example, the left ear of the toy in figure 3 will have approximately the same descriptor as the right ear. If the right ear is matched to it's left counterpart, the matching is on the one hand correct, because an ear has been matched to

In the following we present a matching algorithm that combines n-nearest neighbor matching, transformation clustering and classification to distinguish between features on the object and features on the background in order to calculate an optimal affine transformation matrix that relates the features of the two images.

Let $\mathcal{F}^t$ and $\mathcal{F}^q$ define two sets of features (train and query) with:

$$\mathbf{f}^t_j \in \mathcal{F}^t, \qquad 1 \leq j \leq T = ||\mathcal{F}^t|| \quad \text{and}$$
$$\mathbf{f}^q_i \in \mathcal{F}^q, \qquad 1 \leq i \leq Q = ||\mathcal{F}^q|| \,.$$

Let further denote $d(\mathbf{f}^t_j, \mathbf{f}^q_i)$ the distance between $\mathbf{f}^t_j$ and $\mathbf{f}^q_i$. The set $\mathcal{M}_{\mathbf{f}^q_i}$ of nearest neighbors for $\mathbf{f}^q_i$ is given by:

$$\begin{aligned}
\mathcal{M}_{\mathbf{f}^q_i} &= \left\{\mathbf{f}^t_{j_1}, \ldots, \mathbf{f}^t_{j_K}\right\}, \text{with} \\
j_1 &= \operatorname{argmin}\left\{d(\mathbf{f}^t_1, \mathbf{f}^q_i), \ldots, d(\mathbf{f}^t_T, \mathbf{f}^q_i)\right\} \\
j_2 &= \operatorname{argmin}\left\{d(\mathbf{f}^t_1, \mathbf{f}^q_i), \ldots, d(\mathbf{f}^t_T, \mathbf{f}^q_i) \backslash d(\mathbf{f}^t_{j_1}, \mathbf{f}^q_i)\right\} \\
&\ldots \\
j_K &= \operatorname{argmin}\{d(\mathbf{f}^t_1, \mathbf{f}^q_i), \ldots, d(\mathbf{f}^t_T, \mathbf{f}^q_i) \\
&\qquad \backslash d(\mathbf{f}^t_{j_1}, \mathbf{f}^q_i), \ldots, d(\mathbf{f}^t_{j_{n-1}}, \mathbf{f}^q_i)\} \quad (16)
\end{aligned}$$

and the set of matches for the entire query $\mathcal{M}^q$ image by:

$$\mathcal{M}^q = \{\mathcal{M}_{\mathbf{f}^q_1}, \ldots, \mathcal{M}_{\mathbf{f}^q_Q}\} \,. \quad (17)$$

We use a delaunay triangulation on the positions $\mathbf{p}^q_i$ of the query features $\mathcal{F}^q$ to determine adjacent features. Each triangle connects three query features and is utilized to calculate an affine transformation $\mathbf{T}$ that maps the query features into the coordinate system of their corresponding features:

$$\mathbf{T} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix}. \quad (18)$$

Since a query feature is matched to K test features we get 2^K transformations for each triangle. The transformations will be used to derive a feature vector $\mathbf{w} = (s, \phi, u, t, \mathbf{v}_t)$(Ljubisa and Sladjana, S., 2006), with the parameters:

- $s = \operatorname{sgn}(a)\sqrt{a^2 + c^2}$ - scaling
- $\phi = \arctan(\frac{c}{a})$ - rotation
- $u = \frac{ab+cd}{a^2+c^2}$ - shear
- $t = \frac{ad-bc}{a^2+c^2}$ - compression
- $\mathbf{v}_t = (t_x, t_y)^T$ - translation

another ear, on the other hand it is incorrect because it is the wrong ear.

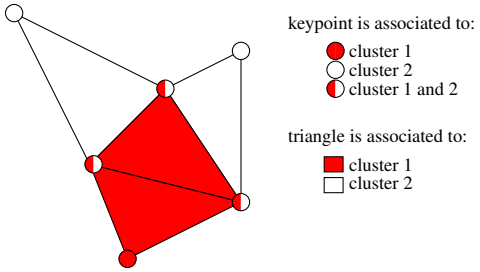


Figure 7: Assignment of each feature to a cluster.

Since the K-nearest neighbor matching introduces many false correspondences it is reasonable to remove the correspondences that are obviously wrong. This could be e.g. done by evaluating the matching distance or utilizing proximity relationships.

To determine those features, that conform to a similar transformation, we normalize each component in $\mathbf{w}$ and use a spectral clustering approach to divide the feature space into two cluster C_1 and C_2.

Afterwards, we use a backprojection algorithm and assign each feature to at least one of the clusters[2] (cf. figure 7). Let $\mathcal{M}_i$ be the set of features, and $\mathcal{T}_i$ the set of transformation matrices, that belong to $\mathcal{C}_i$. The average backprojection error e_i for cluster i over all transformation matrices $\mathbf{T}_{i,j} \in \mathcal{T}_i$ is then defined as:

$$e_i = \frac{1}{||\mathcal{T}_i|| \cdot ||\mathcal{M}_i||} \sum_{j \in \mathcal{T}_i} \sum_{p_k \in \mathcal{M}_i} (\mathbf{p}^q_k \mathbf{T}_{i,j} - \mathbf{p}^t_k) \quad (19)$$

with $\mathbf{p}^t_k$ being the, according to equation 16, corresponding point to $\mathbf{p}^q_k$. The cluster representing the object c_{obj} is then defined as the cluster that minimizes the backprojection error:

$$c_{obj} = \operatorname{argmin}\{e_1, e_2\}. \quad (20)$$

Within the cluster, that represents the object, the optimal transformation matrix is then defined as:

$$\mathbf{T}_{opt} = \operatorname*{argmin}_{j \in \mathcal{T}_{obj}} \left\{ \frac{1}{||\mathcal{M}_{obj}||} \sum_{p_k \in \mathcal{M}_{obj}} (\mathbf{p}^q_k \mathbf{T}_{obj,j} - \mathbf{p}^t_k) \right\} \quad (21)$$

4 EVALUATION

To evaluate the performance of the architecture a benchmark has been implemented, which automatically generates synthetic test images and ground-truth data. Using these the performance of the descriptor by itself, as well as the performance of the descriptor used in conjunction with the matching algorithm

[2]Since a feature could belong to more than one triangle a feature may be assigned to more than one cluster.

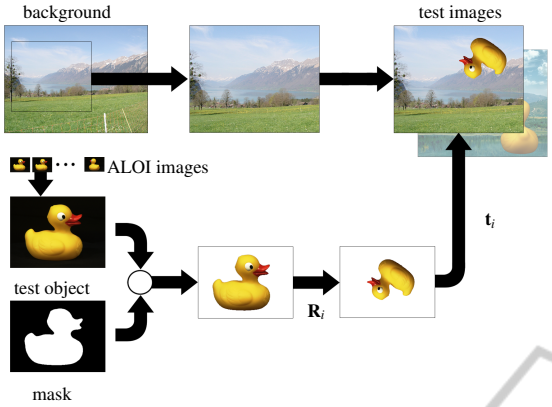


Figure 8: Illustration of the test image generation process.

are evaluated on a textured and non-textured data set. Compared to a test, which uses real images, this approach is able to generate a statistically significant amount of samples.

4.1 Benchmark

The Benchmark uses the *Amsterdam Library of Object Images (Geusebroek et al., 2005)* (ALOI). The ALOI contains images and corresponding masks of a multitude of objects, which were obtained by rotating each object in the z-plane by an angle γ and taking pictures in 5° increments. Hereinafter the process of generating the test images shall be described briefly :

Two angles γ_1 and γ_2 are randomly chosen, based on a threshold γ_{th} with:

$$\gamma_1 - \gamma_2 < \gamma_{th}. \tag{22}$$

By utilizing the images $I_1(\gamma_1)$ and $I_2(\gamma_2)$ a change in perspective is simulated accurately. Afterwards for each image $I_i \in \{I_1, I_2\}$ an angle β_i, which represents a rotation in the image plane; a translation by a vector $\mathbf{t}_i$ and a scaling factor s_i are randomly chosen. Furthermore a rotation matrix $\mathbf{R}_i(\beta_i)$ and an affine transformation matrix $\mathbf{T}_i(\beta_i, s_i, \mathbf{t}_i)$ are calculated. I_i is rotated using $\mathbf{R}_i$, scaled by s_i and inserted at the location $\mathbf{t}_i$ in an output image $\hat{I}_i$. This process is repeated for the object masks yielding the transformed masks I_1^M and I_2^M. In conclusion a randomly chosen background is being added to $\hat{I}_i$. This process is illustrated in figure 8 .

While it is not possible to predict the exact position of the features, since the projection of the features in the image plane is based on the geometry of each object which is rotated by γ, the approximated transformation matrices $\mathbf{T}_{1,2}$ and $\mathbf{T}_{2,1}$ are available. These describe the transformation of the object in the image plane:

$$\mathbf{T}_{1,2} = \mathbf{T}_1^{-1}\mathbf{T}_2 \tag{23}$$

$$\mathbf{T}_{2,1} = \mathbf{T}_2^{-1}\mathbf{T}_1. \tag{24}$$

This approximation holds if γ_{th} is fairly small. Since a big value of γ_{th} would lead to occlusions, which cannot be matched correctly by any approach, this restriction is acceptable.

4.2 Descriptor Evaluation

To test solely the performance of the descriptors, features are extracted for the images $\hat{I}_1$ and $\hat{I}_2$. The resulting sets of features $\mathcal{F}_1$ and $\mathcal{F}_2$ were matched using a brute-force matcher, which employs the L^2 metric, yielding a set of matches $\mathcal{M}_{1,2}$. This process is repeated by switching the query and train feature sets providing a second set of matches $\mathcal{M}_{2,1}$. These matches are of the form $m_k = (m,n), m_k \in M_{i,j}$. m and n describe the index of the matched features $f_m \in \mathcal{F}_i$ and $f_n \in \mathcal{F}_j$ respectively.

Based on these results two metrics are calculated. n_i^{obj} denotes the number of features, that describe the object and were extracted in $\hat{I}_i$. For each feature $f_n \in \mathcal{F}_i$, which is located at $\mathbf{p}_n$, the value of the binary mask I_i^M at $\mathbf{p}_n$ is evaluated, yielding b_n:

$$b_n = \begin{cases} 1, \text{ if } I_i^M(\mathbf{p}_n) = 1 \\ 0, \text{ else.} \end{cases} \tag{25}$$

Therefore:

$$n_i^{obj} = \sum_{n=1}^{||F_i||} b_n. \tag{26}$$

$n_{i,j}^{valid}$ represents the number of correct object correspondences. For each match $m_k = (m,n) \in M_{i,j}$, c_k is calculated:

$$c_k = \begin{cases} 1, \text{ if } I_i^M(\mathbf{p}_m) = 1 \text{ and } I_j^M(\mathbf{p}_n) = 1 \\ 0, \text{ else.} \end{cases} \tag{27}$$

Therefore:

$$n_{i,j}^{valid} = \sum_{k=1}^{||M_{i,j}||} c_k. \tag{28}$$

By conducting this experiment one is able to verify two important aims of this project: The first one is to increase the number of features detected in a non-textured object, which is represented by n_i^{obj}. The second aim is the ability to distinguish these features from those, which belong to the background. This is expressed by the quotient of correct correspondeces and all features on the object:

$$c_{i,j} = \frac{n_{i,j}^{valid} + n_{j,i}^{valid}}{n_i^{obj} + n_j^{obj}}. \tag{29}$$

The evaluation described in this section, was performed on a number of textured and non-textured objects utilizing the SIFT descriptor; the combination of

Table 1: Sum of n_i^{obj}, $n_{i,j}^{valid}$ and mean of $c_{i,j}$, $\bar{c}$ for the textured and non-textured object set.

	textured			non-textured		
	SIFT	form	SIFText	SIFT	form	SIFText
n^{obj}	5093	6312	12054	3720	248	5992
n^{valid}	3973	4622	10044	2450	1943	4844
$\bar{c}$ [%]	78	73	83	66	78	81

MSER and ART as a descriptor (form) and the architecture, which is proposed in this paper. The sums of n_i^{obj},$n_{i,j}^{valid}$ and the mean of $c_{i,j}$, which were obtained during this experiment, are shown in table 1.

The number of detected features by SIFText is significantly higher than the number of features extracted by each partial descriptor. Furthermore the number of correct correspondences, obtained by employing SIFText, exceeds the sum of those generated using the other descriptors. This increase is based on the weighted combination of the descriptor parts and would be lost, if a simple thresholding operation was used instead. Moreover, the values of c calculated on the textured and non-textured data set, while using SIFText, are almost identical. This implies that the descriptor's performance is similar for both datasets; hence it can be utilized effectively for the description of textured and non-textured objects.

4.3 Matching Algorithm Evaluation

Considering the beforementioned ambiguity (cf. section 3) a second test was proposed, which is able to determine the accuracy of the calculation of an object's position and orientation in an image. According to the procedure described in section 4.1 and 4.2 two test images are created. For each test image $\hat{I}_i$ a set of features $\mathcal{F}_i$ is extracted. Based on two sets of features $\mathcal{F}_i$ and $\mathcal{F}_j$ an algorithm akin to the algorithm proposed in chapter 3 is applied, yielding a transformation matrix $\hat{\mathbf{T}}_{\mathbf{i,j}}$. By using equation 25, a subset of features $\mathcal{F}_i^{obj} = \{f_n \in \mathcal{F}_i : b_n = 1\}$ is obtained for each image $\hat{I}_i$, consisting of the features, which are located at the object's location in said image. For each set of features $\mathcal{F}_i^{obj}$ the corresponding oriented bounding box $\mathcal{B}_i$ is calculated and transformed by $\hat{\mathbf{T}}_{\mathbf{i,j}}$ and $\mathbf{T}_{\mathbf{j,i}}$, yielding $\hat{\mathcal{B}}_i$ and $\tilde{\mathcal{B}}_i$. By intersecting the polygons defined by $\hat{\mathcal{B}}_i$ and $\tilde{\mathcal{B}}_i$ the area of the intersection, A_{tp} is obtained:

$$A_{tp} = ||\tilde{\mathcal{B}}_i \cap \hat{\mathcal{B}}_i||. \tag{30}$$

This area represents the portion of the object, which is correctly transformed. The areas A_{fp} and A_{fn} in which only one polygon exists, and thus are falsly matched, are obtained, using the area $\tilde{A}_i$ and $\hat{A}_i$ of $\tilde{\mathcal{B}}_i$ and $\hat{\mathcal{B}}_i$

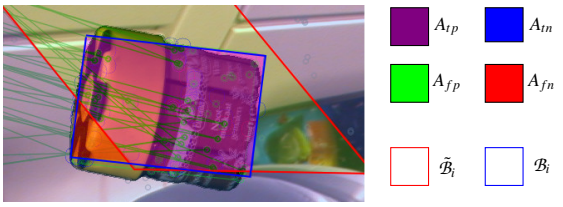


Figure 9: Geometrical interpretation of A_{tp},A_{fp},A_{tn} and A_{fn}, based on $\mathcal{B}_i$ and $\tilde{\mathcal{B}}_i$.

respectively:

$$A_{fp} = \tilde{A}_i - A_{tp} \tag{31}$$

$$A_{fn} = \hat{A}_i - A_{tp}. \tag{32}$$

The portion of the image that is correctly labeled as background A_{tn} is calculated according to equation 33 using the area of the whole test image A_{image}:

$$A_{tn} = A_{image} - A_{tp} - A_{fp} - A_{fn}. \tag{33}$$

A geometrical representation of these calculations is given by figure 9.

Based on the values of A_{tp}, A_{fp}, A_{tn} and A_{fn} confusion matrices were created for the textured and non-textured datasets utilizing a variety of combinations of descriptor and matching algorithms according to the following scheme:

	object	background
object	$\frac{A_{tp}}{A_{tp}+A_{fp}}$	$\frac{A_{fp}}{A_{tp}+A_{fp}}$
background	$\frac{A_{fn}}{A_{tn}+A_{fn}}$	$\frac{A_{tn}}{A_{tn}+A_{fn}}$

Values on the principal diagonal approaching 1 indicate a nearly perfect result, while values close to 0 suggest that the object is detected at the wrong location. The resulting confusion matrices are presented in figures 10 and 11.

In figure 10 the performance of SIFT and SIFText is similar for small values of K , if the experiment is conducted on the textured dataset. Increasing K yields no significant rise in the precision if SIFT is used, while utilizing SIFText significantly increases the value. The afore-mentioned ambiguities are solved, by considering a higher amount of neighbours and the performance is greatly increased. If the non-textured dataset is used, SIFText clearly outperforms SIFT regardless of K. In figure 11 very low precision values are present. Since the background takes up the majority of the image, a correct transformation matrix cannot be deducted from the preponderance of the features. RANSAC isn't suited to be employed in this kind of experiment. Nevertheless this experiment shows that SIFText outperforms SIFT using both datasets.

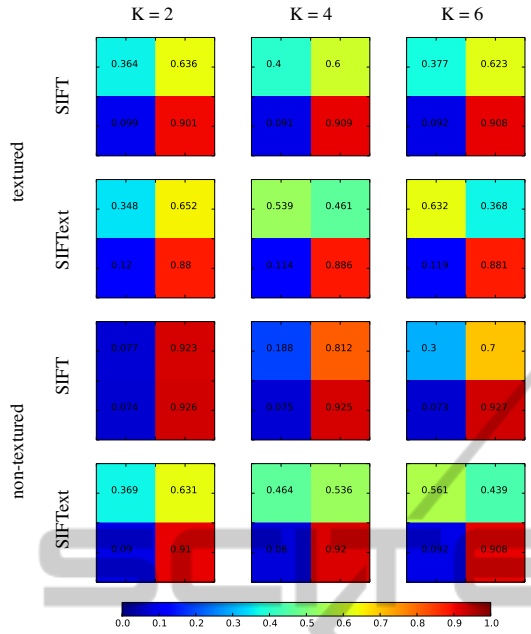


Figure 10: Confusion matrices generated using the algorithm proposed in chapter 3, for various values of the parameter K. As descriptor algorithm SIFT and SIFText were used.

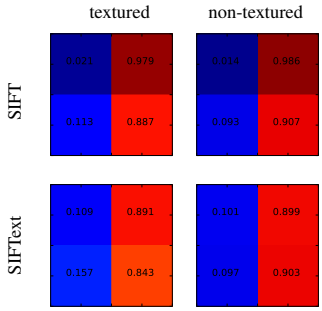


Figure 11: Confusion matrices generated using RANSAC to calculate the homography. As descriptor algorithm SIFT and SIFText were used. The color coding is the same used in figure 10.

5 CONCLUSIONS

In this paper we have implemented and tested a descriptor architecture, which was designed to detect non-textured objects as well as textured objects, by combining edge, shape and color description techniques and weighting them by a locally calculated value, derived from the texture of the image. Furthermore, a matching algorithm was implemented, that was designed to mitigate the weak points of the descriptor. In table 1 we have shown that for each object the number of detected features was increased significantly compared to SIFT, while the number of correct correspondences was increased as well. This applies to both, the textured and non-textured dataset. We have further shown, that the precision of detecting an object's position and orientation in two images is increased compared to RANSAC, if the matching algorithm, which was developed here, is used (cf. figure 10 and 11). But even in the case RANSAC is used, the descriptor surpasses SIFT.

By choosing the weighting factor locally and adapting the descriptor accordingly features that were matched incorrectly by the partial descriptors are assigned correctly. This architecture is, therefore, well suited to detect textured and non-textured objects, as well as objects, that possess textured and non-textured parts.

REFERENCES

Bay, H., Tuytelaars, T., and van Gool, L. (2006). *SURF: Speeded Up Robust Features*. *Computer Vision ECCV 2006*, 3951:404–417.

Bober, M. (2001). Mpeg-7 visual shape descriptors. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6):716–719.

Bulla, C. and Hosten, P. (2013). Detection of false feature correspondences in feature based object detection systems. In *Proceedings of International Conference on Image and Vision Computing New Zealand*, Wellington, New Zealand.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395.

Forssen, P.-E. and Lowe, D.G. (2007). Shape descriptors for maximally stable extremal regions. In *Proceedings of IEEE 11th International Conference on Computer Vision*, pages 1–8.

Geusebroek, J.-M., Burghouts, G.J., and Smeulders, A.W.M. (2005). The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112.

Ljubisa, M. and Sladjana, S. (2006). Orthonormal decomposition of fractal interpolating functions. *Series Mathematics and Informatics*, 21:1–11.

Lowe, D.G. (2004). *Distinctive Image Features from Scale-Invariant Keypoints*. *International Journal of Computer Vision*, 60(2):91–110.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761 – 767.

Naik, S. and Murthy, C. A. (2007). Distinct multicolored region descriptors for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1291–1296.

Tombari, F., Franchi, A., and Di Stefano, L. (2013). Bold features to detect texture-less objects. In *Proceedings*

of IEEE International Conference on Computer Vision, pages 1265–1272.

Toshev, A., Taskar, B., and Daniilidis, K. (2012). Shape-based object detection via boundary structure segmentation. *International Journal of Computer Vision*, 99(2):123–146.

Van De Weijer, J. and Schmid, C. (2006). Coloring local feature extraction. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part II*, pages 334–348.