8

Self-Calibration of Wide Baseline Stereo Camera Systems for Automotive Applications

Nico Mentzer[1], Guillermo Payá Vayá[1], Holger Blume[1], Nora von Egloffstein[2] and Lars Krüger[2]

[1]Institute of Microelectronic Systems, Leibniz Universität Hannover, Hannover, Germany
[2]Daimler AG, Vision Enhancement, Ulm, Germany

8.1 Introduction

Many car accidents involving vulnerable road users (e.g., pedestrians or cyclists) occur on rural roads after dark, when the driver's visibility is restricted. Thus, the main objective of an augmented night vision is to assist the driver, when driving on side roads (e.g., highways, country roads, or rural roads) with poor or restricted visibility by alerting the driver to potential obstacles ahead.

One possible augmentation of driver vision is to highlight potential obstacles, hazards or vulnerable road users in the live video of the road ahead. A classification of image content is mandatory for this application. As the augmentation enables the driver to grasp the situation quickly, the distance to the detected object has to be calculated by stereo vision to ensure accuracy and speed of assessment.

As the range of distance resolution increases with the baseline of a stereo system, a wide baseline stereo system is necessary to facilitate the augmentation of objects in the desired range. Such a wide-baseline stereo system is sometimes not practicable when rigidly coupled, therefore cameras are mounted individually, e.g., to the windshield. Physically separated cameras increase the camera baseline, however a moving car causes multiple vibration sources [1] which misalign the images of the separated cameras. Therefore, online camera calibration is indispensable for further image processing. This online camera calibration covers the reconstruction of extrinsic camera parameters, which rely on a sparse pixel correspondence list from the two camera images. The general overview of the algorithmic flow is depicted in Figure 8.1. This chapter will focus on the search for sparse pixel correspondences and extraction of camera calibration parameters.

The remaining chapter is set up as follows. Section 8.1 gives an introduction to the self-calibration of wide baseline stereo cameras. After a review of the considered algorithms in Section 8.2, Section 8.3 details the class of image feature detectors and extractors. Section 8.4 highlights the matching of image features. An in-depth description of the bundle adjustment for the camera calibration is given in Section 8.5. In Section 8.6, selected application-specific aspects regarding the algorithmic parameterization are presented. Section 8.7 focuses on algorithmic-specific and hardware-specific implementation details and gives an overview of existing implementations for the extraction of image features.

8.1.1 Extraction of Image Features

Image feature extraction consists of two steps: the detection of image features and the generation of the descriptor for those feature points, which results in a unique signature as a representation for the detected feature points.

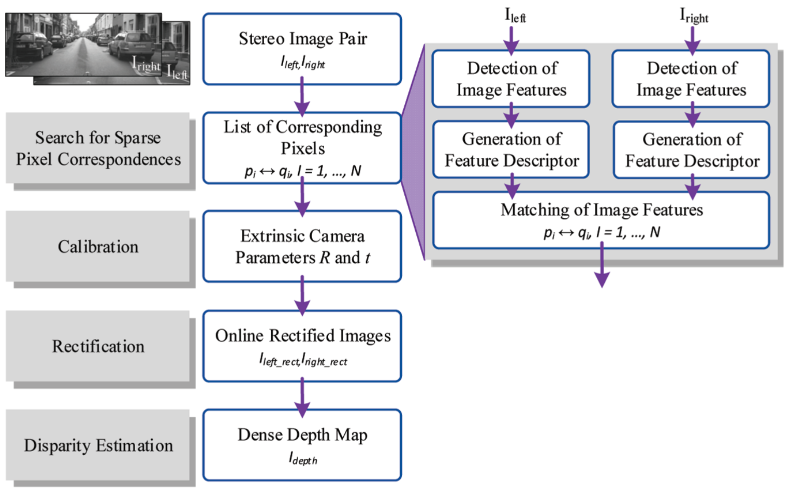

I_right I_left
Stereo Image Pair
I_{left}, I_{right}
Search for Sparse Pixel Correspondences
List of Corresponding Pixels
$p_i \leftrightarrow q_i, l = 1, \ldots, N$
Calibration
Extrinsic Camera Parameters R and t
Rectification
Online Rectified Images
$I_{left\_rect}, I_{right\_rect}$
Disparity Estimation
Dense Depth Map
I_{depth}
I_{left}
I_{right}
Detection of Image Features
Detection of Image Features
Generation of Feature Descriptor
Generation of Feature Descriptor
Matching of Image Features
$p_i \leftrightarrow q_i, l = 1, \ldots, N$

Figure 8.1 Algorithmic overview. Input of the processing chain is a stereo image pair, in which sparse pixel correspondences are extracted for online camera calibration. After the calibration, rectification is performed as a preprocessing step for disparity estimation.

The image feature detection generates a list of distinctive invariant points in images for the feature localization. Especially for camera calibration, a high accuracy of localization is required [2] in order to ensure a correct functionality of following algorithmic steps, e.g., the rectification of stereo image pairs. Due to the similarity between the views of the scene, a rotation invariance or scale invariance of the feature descriptors supports stability of the matches. This is however, not mandatory, because characteristic points in image pairs of the used stereo camera configuration rarely change their rotation or scale abruptly from left to right stereo image.

In recent years, three different principles for **feature detection** have proven employable. *Corner* or *edge detectors* extract characteristic corners or edges in an image, which are defined by large gradient changes of image intensities. So called *blob detectors* determine pixel positions, for which a circular local neighborhood is approximately constant or similar for a defined image property [3]. Furthermore, *affine invariant detectors* have been adapted to be invariant to affine transformations, which are approximations to perspective distortions in order to achieve invariance to large changes in viewpoint [4]. The detected features of the exemplary SIFT-feature detector are shown in Figure 8.2.

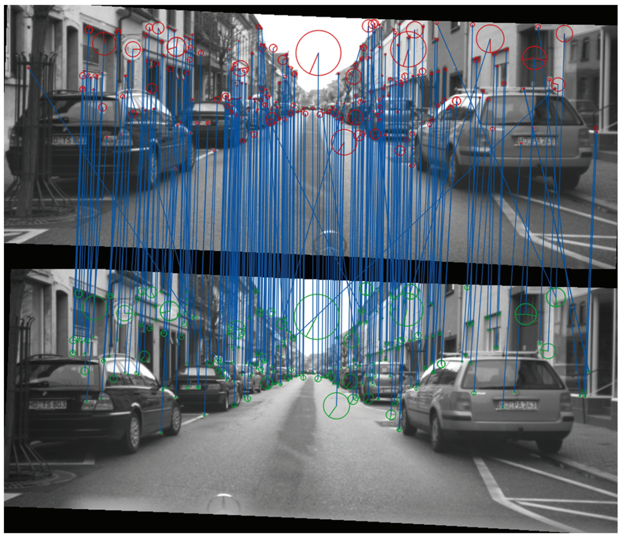


Figure 8.2 Left (*top*) and right (*bottom*) image from a stereo camera system showing detected SIFT-image features. Detected feature points of the left/right image are displayed in red/green, matches are displayed in blue. Scale and rotation of the SIFT-features are illustrated by the circle properties.

The descriptor of an image feature characterizes the detected feature point. Ideally, a **feature descriptor** of a world point is unique when compared to other descriptors, but identical for the same world point in different views [5]. Two representations for descriptors have been established in recent years. So called *histogram-based or distribution-based descriptors* represent the local neighborhood of a feature point by histograms of local image properties like pixel intensities, color, texture, edges etc. [3]. Furthermore, *binary descriptors* represent a local pixel region by storing the binary result of predetermined pixel-level intensity comparisons [6]. In contrast to distribution-based descriptors, binary descriptors contain a more compact representation of the image patch around a feature.

In general, extracted image features have to cope with various influences. Firstly, there are disruptive effects related to the image quality, e.g., image compression, image noise, image blur due to zoom or exposure. Secondly, there are influences resulting from the content of the stereo image pair, e.g., illumination, difficult viewpoint conditions or occlusions, background clutter and general content changes, perspective changes or changes in the view point of planar and non-planar geometry [6, 7]. Finally, application specific factors as scale and rotation of objects impact the algorithmic results dealing with image features. Thus, extracted image features have be invariant to as many disturbing influences of the named categories as possible.

The large variety of image feature detectors and descriptors clearly show the manifold approaches to defining and describing characteristic points in images. As S. Gauglitz mentioned before in [5], "there is no clear-cut definition as to what makes a point interesting. Detection of such points is only an intermediate step in any application". There is no general answer for the question, which detector or descriptor is performing the best. Therefore, as J. Shi and C. Tomasi postulated in 1994, "the right features are exactly those that make the tracker work best" [8]. Consequently, "any set of feature points is acceptable, but the result ought to be consistent, e.g., in images that show the same scene, the algorithm should detect the same points." [5]. In other words, for each application, the best performing combination of image feature detector and extractor has to be found. Furthermore, application-specific conditions (here: high localization accuracy with low requirements to scale and rotation invariance) aggravate the possibilities of algorithmic combinations.

A survey of existing image feature detectors and descriptors will be given in Section 8.2. A more detailed presentation of an exemplary feature detector and descriptor called SIFT (Scale-Invariant Feature Transform) [9], which shows good results in this application, will be given in Section 8.3.

8.1.2 Matching of Image Features

Matching image features results in a list of pixel correspondences between the left and right input image of the stereo image pair. The main challenge is on the one hand to find as many corresponding pixels as possible while avoiding wrong pixel assignments on the other, even if there are several similar regions in both input images. The assignment of image features to pixel correspondences is based on feature descriptors, which are used to find the maximum similarity between the extracted image features. Depending on the representation of the features (histogram-based or binary descriptor), the similarity is computed by various vector norms for the distance of two matching candidates or the Hamming distance. Furthermore, different matching methods have a significant impact on the resulting correspondence lists [3].

In the case of global feature matching methods f : $\widetilde{X} \rightarrow \widetilde{Y}$, two feature points $\overrightarrow{x} \in \widetilde{X}$ and $\overrightarrow{y} \in \widetilde{Y}$ are assigned by local similarity, which is determined by the related descriptors $\overrightarrow{d_x}$ and $\overrightarrow{d_y}$. For each descriptor in set Y, there is a corresponding descriptor in set X with a minimal error criterion. After the assignment of feature points, the correspondences are filtered by this error criterion in order to avoid false correspondences, e.g., feature points which are not detectable in both images because of occlusions in one image. Varying matching methods differ in the error criterion for the evaluation of feature similarity and the search algorithm during the matching step.

8.1.3 Extrinsic Online Self-Calibration

Common stereo algorithms for disparity estimation (e.g., [10]) rely on exact knowledge about the intrinsic (e.g., focal length) and extrinsic camera parameters (the transformation between two cameras). Calibration errors lead to erroneous reconstruction values. The camera parameters enable the rectification, which is the projection of the camera images to a common image plane and they form the basis for further processing.

The intrinsic parameters may be assumed to be constant and identified using an offline calibration procedure (e.g., [11]). As the cameras are not rigidly coupled here, the extrinsic parameters vary due to vibrations in the car and are assumed to change rapidly from frame to frame. Thus, a one-time offline calibration procedure does not suffice to meet the accuracy requirements of stereo processing. Thus, an online calibration procedure is necessary. While driving the use of calibration targets with known geometry is difficult. Therefore, a self-calibration mechanism is needed.

The idea behind online self-calibration procedures is to estimate the camera parameters based on what is perceived in both cameras. So a preprocessing step to the calibration is a one-to-one identification of scene points visible in both camera images, e.g., a list of sparse pixel correspondences of the stereo camera images.

8.2 Algorithmic Overview

Many approaches have been proposed in recent years for the extraction and matching of image features and for the feature-based camera self-calibration. In the following section, selected aspects for each algorithmic step are reviewed separately.

8.2.1 Survey of Image Features Extraction

The process of extracting image features is split into two algorithmic parts, the detection of feature points and the generation of the feature descriptor. For both steps, a large number of algorithms have been published. In this section, typical examples of each algorithmic step are presented.

8.2.1.1 Detection of features

Which properties of distinctive image points are mandatory for a satisfactory matching of image features depend on the finale application. There is no clear definition as to which extraction strategy is best as it only needs to provide sufficient algorithmic performance during retrieval in the same scene on image sequences from different viewpoints. Therefore, what is *characteristic* for highly distinctive points in images is an application-specific approach, which has led to four basic methods for extracting retrievable points in images.

Edge detection

Edges are stable features, which are detectable over a range of viewpoints and illumination changes [12]. An edge, e.g., the border of an object, is defined by discontinuities in pixel intensities in a single image dimension (see Figure 8.3(b)). Thus, the Canny detector [13] determines the gradient of the input image with the Sobel operator and by evaluating magnitude and orientation of the gradients, the edge's direction and its strength are extractable.

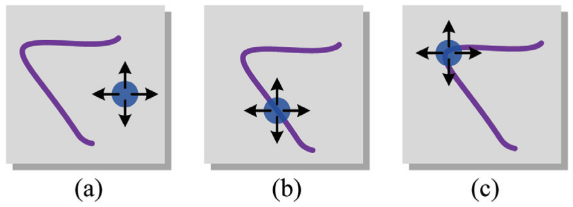


Figure 8.3 Detection of edges and corners by image gradients. The blue circle shows a possible feature point, surrounded by a local neighborhood. (a) Low image gradients in two spatial directions represent texture free image areas. (b) A high image gradient in one spatial direction indicates a possible edge, (c) in two spatial directions a possible corner.

Gradient and direction are used in a non-maximum suppression in order to suppress equivocal edges in the local neighborhood of a possible edge.

The drawback of this method is the equivocalness of the detected feature points. As depicted in Figure 8.3(b), it is not distinct which detected points are corresponding on the edge while matching two detected feature points and therefore, it will lead to incorrect pixel correspondences.

Corner detection

Corners are defined as intersections of edges or as pixel continuities in two or more image directions (see Figure 8.3(c)). In addition to simple corners, line endings and cropped intensity changes are detected using this type of detector.

One early corner detector is the Harris corner detector [14] (1988), which approximates the sum of squared differences of two image patches in order to detect a difference in image intensities. The approximation results in the second moment matrix, which represents the dominant directions of a local neighborhood in the gradient image. With this approach it is not only possible to detect corners, but edges as well.

To avoid such costly filters, a detector has been presented that does not rely on discrete image derivatives, but on the number of intensity differences between pixels [5], which are located on a Bresenham circle (see Figure 8.4). Rosten [15] sped up this process by reducing the number of pixel tests with machine learning techniques to find the fastest sequence of pixel comparisons for rejecting a wrong corner candidate.

The matching of detected corners in different images of the same scene provides correct pixel correspondences as long as the detected corners belong to objects of the same size. A corresponding corner is just detectable in different images, if the regions for describing the corners have similar dimensions

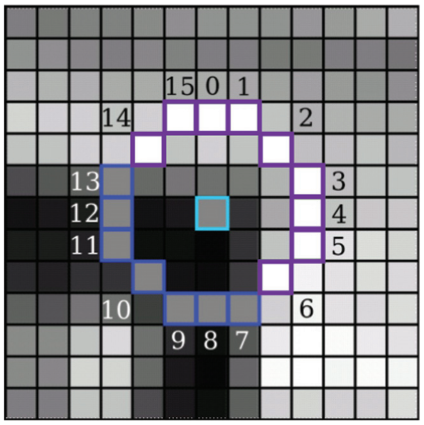


Figure 8.4 Intensity comparisons of pixel, which are located on a Bresenham Circle. The central pixel is determined as a corner if a certain number of continuous pixel intensities is brighter or darker than the central pixel. This is combined with an adoptable threshold to avoid instabilities.

(see Figure 8.5, red circle), which is dependent on the object size. To overcome this problem, repeated image scaling is a possibility or an object size dependent adjustment of the region for the descriptor generation.

Blob detection

A blob is a region of connected pixels, which share a common image property, e.g., pixel intensities, and therefore stand out from surrounding regions. By formulating image properties as a function of pixel positions, local maxima and minima of the function are determinable.

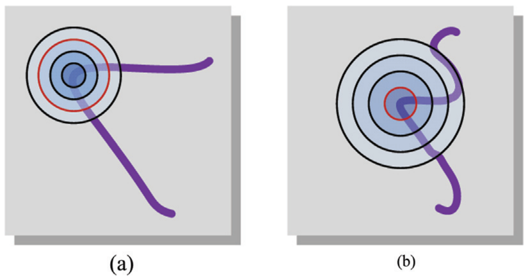


Figure 8.5 Detection of corners of different image scales. With strongly different object sizes in the image, a corresponding corner is not detectable (red circle), but by a repeated image scaling.

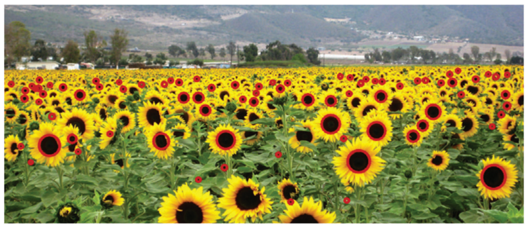


Figure 8.6 Blob detector. The detected blobs are displayed as red circles. The blob's size is displayed as the diameter of the circle.

It has been shown, that the Laplacian of Gaussian (LoG) [16] has a strong response to dark and bright image regions, which are detectable as blobs. The response is highly dependent on the size of the filter kernel used (see Figure 8.6).

Affine-invariant interest point detection

Images features based on a blob detector hardly match for large scale or viewpoint changes [4], because circular image patches for blob feature extraction will lead to large distance measures for blob feature matching due to less covering of the circular regions (see Figure 8.7). By applying circular image patches, the used image information is too different to ensure stable pixel correspondences for large viewpoint changes. Therefore, Mikolajczyk [7] extends blob detectors to affine invariance by estimating the affine shape of a local neighborhood. For affine transformations, the scale of an image region changes differently in each direction, which leads to differing local regions for the blob detection and therefore to differing localization or to mistaken detections.

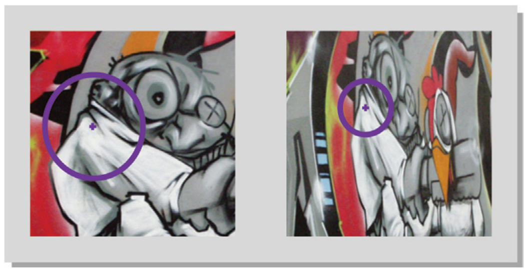


Figure 8.7 Blob detection based on circular image region for a scene with a large viewpoint change. The region on which the blob feature extraction is based only partially covers the corresponding region and thus, will lead to non-matching image features.

In order to deal with affine transformation, Mikolajczyk [7] replaces the blob detection scales, which are equal in all directions, by affine detection scales, which vary independently in orthogonal directions. Hereby, the circular point neighborhood is replaced by an ellipse, which is determined by the second moment matrix. With the affine normalization, the ellipse is normalized to a circle again and a blob is detectable within the transformed image patch (see Figure 8.8).

Since the four presented methods provide large differences in quantity and quality for detected interest points, a suitable algorithm has to be chosen with regards to the application.

In 3D reconstruction, precise localization of interest points is one major aspect [4], therefore a sub-pixel accuracy for feature detection is mandatory. Self-occlusion occurs very frequently in real world scenes and typically many interest points are found near occlusion boundaries. Accurate positioning of features is imperative. As has been shown in many publications, center-oriented detectors (e.g., LoG, DoG or CenSurE) [5], provide a higher and more stable repetition rate than corner or edge detectors. Furthermore, affine-invariant interest point detectors have been adapted to be robust to large changes in viewpoint [4], which is of minor importance even for reliable image feature matching for a wider baseline stereo camera system.

Taking into account the algorithmic robustness of the presented methods for the detection of image features and the high requirements of ADAS (Advanced Driver Assistance Systems), a blob detector is used for the detection of features henceforth. In subsection 8.3.1 the SIFT-detector [9] will be presented in detail as an exemplary blob detector.

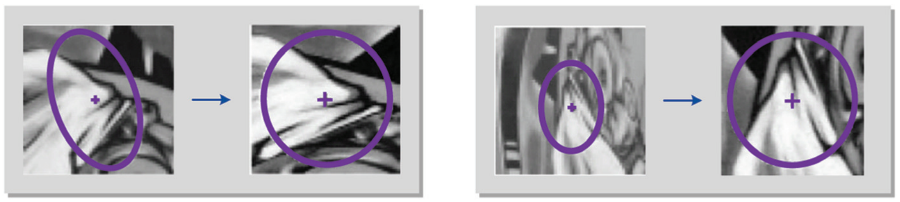


Figure 8.8 Affine-Invariant Interest Point Detection. The circular point neighborhood is replaced with an ellipse in order to achieve independent orthogonal varying detection scales for interest point detection. Before applying a detection algorithm, the local neighborhood is affine normalized, which results in a circular neighborhood and a transformed image patch (from [7]).

8.2.1.2 Description of features

After the detection of interesting points, the descriptor as a unique representation of an image feature has to be generated. In addition to histogram-based descriptors, which are memory greedy, binary descriptors have been established as a more compact representation for image features. In addition, compared to histogram-based descriptors, the distance of two binary descriptors, which is required for feature matching, is faster to match. There are other techniques to describe image features such as image patch correlation or generalized moment invariants [3], however the focus of this section is limited to the two mentioned descriptor types, due to their suitability for the self-calibration of wide baseline stereo camera systems.

Histogram-based descriptors

A simple way to describe a detected blob in a histogram-based manner is the distribution of pixel intensities of the local blob region. Due to the fact that this technique is prone to illumination changes, more complex approaches have been presented (see [3]), e.g., the distribution of gradient locations and orientations in the local blob area instead of the distribution of pixel intensity itself. In the case of the SIFT-descriptor, the coordinates of the descriptor and the gradient orientations are rotated relative to the feature orientation and afterwards, a histogram is generated based on orientation and magnitude of the image gradient [9]. Furthermore, the quantization granularity of gradient locations and orientations leads to a robust descriptor, which is stable to small geometric distortions and small errors in the blob region. Besides multiple techniques for histogram generation, different sampling grids have been introduced (see Figure 8.9). The resulting descriptor is a multidimensional vector with the histogram's bins as components. In the case of SIFT,

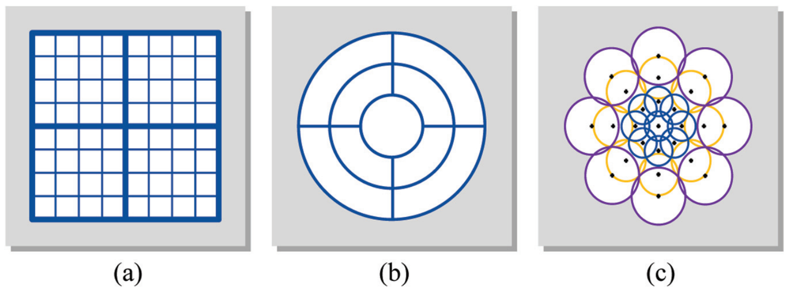


Figure 8.9 Sampling grids for generating different descriptors: (a) SIFT [9], (b) Shape Context [18], (c) DAISY [19].

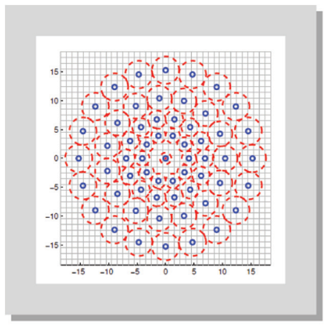
each vector consists of 128 values of floating point precision. The size of a feature vector is highly dependent on the algorithmic parameters, but nevertheless histogram-based descriptors usually have high memory requirements. Therefore, techniques for a more compact descriptor representation have been developed, e.g., principal component analysis for PCA-SIFT [17].

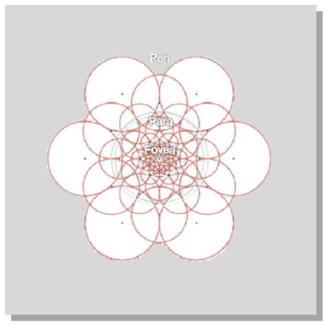
Binary descriptors

Due to the fact that histogram-based descriptors provide a large complexity [3] and high memory requirements [6], a sped up generation and a more compact representation for feature descriptors is desirable. Therefore, binary descriptors are characterized by sampling patterns and predefined sampling pairs. Sampling patterns define a set of potential sampling locations (Figure 8.10, blue circles), whose image information are optionally smoothed with spatial-dependent filter kernels (e.g., Gaussian smoothing) (Figure 8.10, red circles). A fixed combination of the filtered intensities is selected in advance as descriptor specific sampling pairs (see Figure 8.11, two variations of sampling pairs for the FREAK descriptor).

For each sampling pair, a binary test τ is performed, e.g., (BRIEF [20]):

$$\tau(\boldsymbol{p}; x, y) := \begin{cases} 1 & if\ I(\boldsymbol{p}, x) < I(\boldsymbol{p}, y) \\ 0 & otherwise \end{cases}$$


(a)



(b)

Figure 8.10 Sampling pattern. (a) BRISK descriptor, (b) FREAK descriptor [21]. Sampling patterns define a set of sampling locations (blue circles), of whose image information is smoothed with spatial-dependent filter kernels (red circles). Out of the sampling pattern the sampling pairs for the binary tests for the descriptor generation are selected.

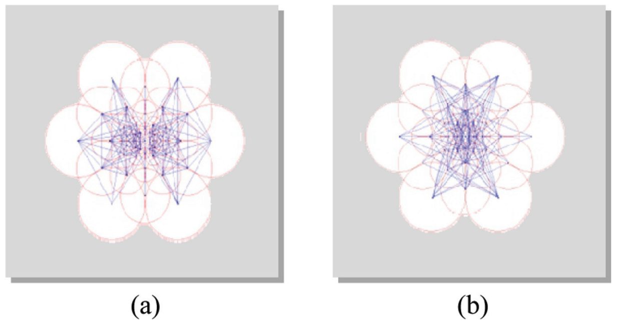


Figure 8.11 Two variations of sampling pairs of the FREAK descriptor [21]. A fixed combination of sampling locations is selected as descriptor specific sampling pairs, with which the binary tests for the descriptor generation is performed.

where $I(\boldsymbol{p}, x)$ is the pixel intensity in a smoothed image patch $\boldsymbol{p}$ around an image position $x = (u, v)^T$. On a set of n_d precomputed pixel pairs, such binary tests are performed. The resulting descriptor of dimension n_d ensues to

$$\sum_{1 \le i \le n_d} 2^{i-1} \tau(\boldsymbol{p};; x_i, y_i)$$

Typically, a binary descriptor has a maximal length of 512 Bit.

8.2.1.3 Characteristics of features

Invariances to rotation and scale increase the detection rate of features in similar views of a scene and ensure the distinctiveness of the detected feature points. By assigning a region based main orientation, a feature is rotated by this orientation in order to match it with a corresponding feature from a different orientation. Furthermore, objects often vary in size in different images, which lead to variant image regions for the description of the same feature. To unify the descriptor generation, Lindeberg's [16] scale-space theory is applied.

Rotation invariance of a feature descriptor is achieved by rotating the sampling grid or sampling pattern for the pixel area which is used for the descriptor generation by the main orientation before the descriptor is extracted (see Figure 8.12) or by rotating the descriptor itself. To determine the main orientation, different approaches are available. Rublee et al. [22] use intensity centroids to determine the main orientation of a patch, whereas Leutenegger et al. [23] use the gradient of predefined sampling pairs to rotate the sampling pattern. Further techniques are available in the literature (e.g., [9, 21, 24]).

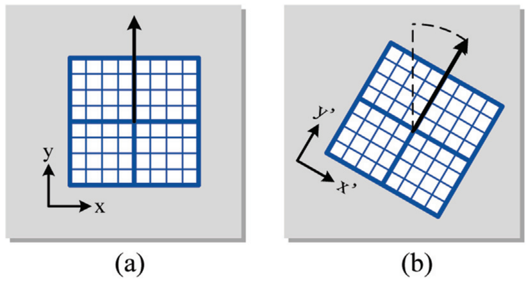


Figure 8.12 Rotation invariance is achieved by rotating the sampling grid by the main orientation before extracting the descriptor.

Scale invariance of image features is attained by applying Lindeberg's [16] scale-space theory for image processing to the input images while detecting image features. The input image is subsampled multiple times to generate different scales of the input image and the detection step is repeated. If the same feature candidates are detected on multiple scales, the candidate on the scale with the highest information content is selected in order to achieve scale-invariance (see Figures 8.13 and 8.14). Lowe (SIFT, [9]) approximates

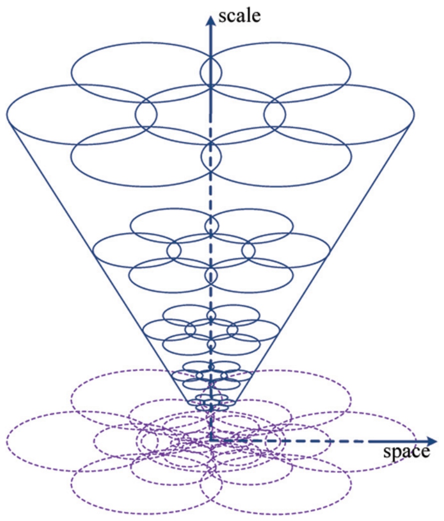


Figure 8.13 Scale-space. An input image is down sampled to achieve multiple scales of the image. On each scale, feature candidates are found, whereas repeated candidates are removed. The scale with the highest information content for the feature candidate is selected as the feature scale (from [16]).

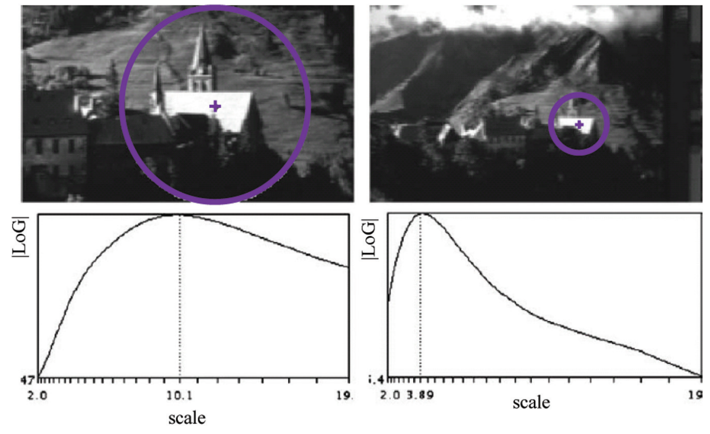


Figure 8.14 Multi-scale approach for blob detection. The same blob with differing scales in two images and the related response (normalized Laplacian of Gaussian) over scales is shown. The scale with the highest information content is chosen as a blob (from [7]).

Lindeberg's LoG scale-space with different Gaussian smoothed images and therefore, the complexity is reduced significantly.

A further approach for scale invariance is the detection and later suppression of feature candidates which are detected on multiple scales, but have the same image position. Those repeated nominations are compensated by a non-maximum suppression [6], which evaluates a predefined cornerness score and selects the most unique feature point.

Image feature detection and description are not completely independent. By choosing a certain feature detector, a specific local neighborhood is used to detect interesting points. This specific local neighborhood has to be also employed to extract the feature descriptor in order to ensure a reliable description of the image patch. Although it seems to be a promising approach, it is not advisable to combine any detector with any descriptor [4]. The following overview (see Tables 8.1 and 8.2) of selected state-of-the-art feature extractors and feature descriptors with references is not intended to be exhaustive, but gives an impression of how many different detectors and extractors are available and therefore combinable. For an appropriate performance, each algorithm requires an application-specific parameterization, which may depend on the previous and following processing step. Thus, this large number of degrees of freedoms results in an algorithmic variety, which is hardly ascertainable.

Table 8.1 Overview of feature detectors

Feature Detector	Year	Comment
SIFT [9]	1999	Scale-Invariant Feature Transform
		Scale-space based, invariant to scale and rotation
SURF [25]	2008	Speeded Up Robust Features
		Scale-space based, invariant to scale and rotation
KAZE [24]	2012	Non-linear scale-space based
		Invariant to scale and rotation
A-KAZE [26]	2013	Accelerated-KAZE
		Improved KAZE feature detector
BRISK [23]	2011	Binary Robust Invariant Scalable Keypoints
		Scale-space based, invariant to scale and rotation
FAST [15]	2006	Features from Accelerated Segment Test
		Segment based corner detector
ORB [22]	2011	Oriented FAST and Rotated BRIEF
		Advanced from FAST and BRIEF (see descriptors)

Table 8.2 Overview of feature descriptors

Feature Descriptor	Year	Comment
SIFT [9]	1999	Scale-Invariant Feature Transform
		Histogram-based descriptor
SURF [25]	2008	Speeded Up Robust Features
		7 Histogram-based descriptor
KAZE [24]	2012	Non-linear scale-space based
		Histogram-based descriptor
A-KAZE [26]	2013	Accelerated-KAZE
		Binary descriptor
BRISK [23]	2011	Binary Robust Invariant Scalable Keypoints
		Binary descriptor
BRIEF [20]	2012	Binary Robust Independent Elementary Features
		Binary descriptor
ORB [22]	2011	Oriented FAST and Rotated BRIEF
		Advanced from FAST and BRIEF (see detectors)
DAISY [19]	2010	Dense Descriptor for Wide Baseline Stereo Matching
		Histogram-based descriptor
FREAK [21]	2012	Fast Retina Keypoint
		Binary descriptor

8.2.2 Feature Matching

The final step in finding sparse pixel correspondences is the assignment of the extracted image features in different image set ups, e.g., in time sequentially images for sparse optical flow, in stereo image pairs for feature-based sparse disparity estimation or in image patches for object detection.

As in the case of the previous algorithmic steps, many approaches for descriptor matching have been presented in recent years [3]. In order to determine the similarity of two image features, multiple correspondence measures are available. In addition, various matching methods lead to significant differences in matching results, which influences the resulting pixel correspondence lists and finally, some matching methods require a list search algorithm, for which again different approaches are available. Each aspect will be briefly reviewed in the following subsection.

Correspondence measures for image features

For histogram-based descriptors $\overrightarrow{d} \in \mathbb{R}^l$, which are real-valued vectors of dimension $l \in \mathbb{N}$, multiple vector norms are applicable on matching difference vectors as a similarity measure. The sum norm is defined as the accumulation of the component wise sum of absolute differences:

$$\|\overrightarrow{d}_x - \overrightarrow{d}_y\|_1 = \sum_{i=1}^{l} |d_{x,i} - d_{y,i}|$$

In order to weight large vector difference more than small differences, the Euclidean norm is useable. The norm penalizes large vector differences more than small vector differences by accumulating the component wise sum of squared differences:

$$\|\overrightarrow{d}_x - \overrightarrow{d}_y\|_2 = \sqrt{\sum_{i=1}^{l} |d_{x,i} - d_{y,i}|^2}$$

Since only relative correspondence measures are used for feature matching, the square root is skippable to avoid costly computations.

A further method for evaluating the distance of two vectors is the normalized cross correlation:

$$distance = max_{x \in X} \left(\frac{\sum_{i=1}^{l} d_{x,i} \cdot d_{y,i}}{\sqrt{\sum_{i=1}^{l} d_{x,i}^2} \cdot \sqrt{\sum_{i=1}^{l} d_{y,i}^2}} \right)$$

The correlation yields good results for the matching of image features, but leads to high computational complexity [3] and is therefore rarely used for matching of image features in the field of advanced driver assistance systems.

For binary descriptors, which consist of a bit string of length n, that represent the result of pixel wise test, the correspondence measure is the

Hamming distance, which is the accumulation of the bit wise XOR of the bit strings:

$$ham_{\overrightarrow{d}_x, \overrightarrow{d}_y} = \sum_{i=1}^{n} (d_{x,i} \oplus d_{y,i})$$

Due to the correspondence measure's simplicity, typically the distance computation of two binary descriptors is noticeably faster than the distance computation of two histogram-based descriptors. Contrary, not every binary descriptor has a comparable quality level as histogram-based descriptors for certain applications. By selecting a specific descriptor type, the implicit trade-off between execution time and descriptor quality has to be taken into account.

Matching methods for image features

The quality of resulting pixel correspondences highly depends on the utilized matching method. Three different methods have been established in the field of feature matching for advanced driver assistance systems (from [3]), which show different behavior in the matching inlier/outlier ratio:

1. **Threshold-Based Matching (TB)**
   Two features match, if the distance between the descriptors is below a predetermined threshold. A feature may have several matches and several of them may be correct.
2. **Nearest-Neighbor-Based Matching (NNB)**
   Two features match, if the descriptor $\overrightarrow{d}_y$ is the nearest neighbor to $\overrightarrow{d}_x$ and if the distance between the descriptors is below a threshold. A feature only has one match
3. **Nearest-Neighbor Distance Ratio Matching (NNDR)**
   Two features match, if the descriptor $\overrightarrow{d}_y$ is the nearest neighbor to $\overrightarrow{d}_x$ and if a ratio ε between the first and the second nearest neighbor is below a threshold:

   $$\varepsilon = \frac{\|\overrightarrow{d}_x - \overrightarrow{d}_y\|_p}{\|\overrightarrow{d}_x - \overrightarrow{d}_z\|_p}$$

   where p indicates the type of norm. This ratio avoids ambiguous matches in case there are potential matches with a similar distance. Again, a feature has only one match.

The matching quality for both nearest-neighbor approaches are higher than for the TB matching [3], because the probability of a correct match for the nearest neighbor matchings is higher than the TB matching, although the distance between similar descriptors possibly varies significantly. The nearest neighbor matchings select only the best match below the threshold and rejects all others and thus, there are few false matches. In addition, the NNDR matching penalizes descriptors which have many similar matches, e.g., the distance to the nearest neighbor is comparable to the distance of the second nearest neighbor. This leads to further improvement in precision. The drawback of the nearest neighbor matchings is the complexity when matching two large pools of image features and the computative costly division for the NNDR matching.

List search approaches for matching of image features

The matching of two large pools of image features to find pixel correspondences in different images results in a costly process, because a correspondence measure and the first two nearest neighbors have to be evaluated for each possible feature combination. By restricting the pool of feature candidates for the matching process, a significant reduction of problem size is achievable. A possible restriction bases on feature properties, e.g., localization in the image, orientation or scale. Constraining the feature candidates means, that the pool of all image features has to be scanned for valid candidates, which is a list search problem.

1. **Sorted Linear Candidate Search**
   A prior sort of the pool regarding the restriction parameter enables a reduction in search time. By using the iterative successively approximation, the list index of the first element which fulfills the restriction is searched. The last candidate of the reduced list is searched with a linear search.
   After each iteration, the step size is halved and the search index is incremented or decremented depending on whether the restriction criterion is fullfilled. The initial step size is half the initial pool size.
2. **KD-Tree Candidate Search**
   A KD-tree [27] based search is a search tree with two edges per vertex and which divides the remaining set of feature candidates into two sets of the same size. By stepping through the KD-tree, the index of the first valid feature candidate is found efficiently. The disadvantage of this search method is the time consuming *a priori* construction of the KD-tree, which is not effective for small feature pools. In addition, if the

restriction search space has a low dimension, other search methods will perform faster.

8.2.3 Survey of Feature-based Self-Calibration

Extrinsic camera self-calibration is about recovering the extrinsic camera parameters using scene point correspondences only. Camera self-calibration is still a wide field of active research with different approaches. Early approaches are subdivided into aiming 3D reconstruction or not. The latter covers those algorithms where no information about the scene in front of the cameras is recovered during optimization.

One of the first approaches has been proposed by Longuet-Higgins [28]. The author introduced a linear method to recover the essential matrix, which is decomposable into the extrinsic parameters. Due to the required number of image point correspondences, it was introduced as the 8-point-algorithm.

Several following publications proposed optimizations regarding decomposition [29], plausibility [30, 31], and outlier handling for the corresponding image points [32]. As the linear approaches often lack the required accuracy, they are often followed by a non-linear refinement in a stratified process.

On the other hand, there are algorithms where camera parameters and 3D points of the scene are recovered simultaneously. One of those is bundle-adjustment [33]. Here a good initialization is required as Gauss-Newton optimization is involved. Thus, bundle adjustment is often chosen for the non-linear refinement as mentioned before.

Regarding online calibration procedures, they are classifiable as recursive or non-recursive. Recursive, or continuous self-calibration, means that temporal constraints are also optimized. Thus, image measurements in earlier time steps influence the current calibration result. Dang et al. proposed a parameter tracking system involving epipolar constraints and bundle adjustment [34]. In contrast to non-recursive self-calibration, there are no temporal constraints. Those are applied, in cases of a continuous decalibration or for active systems. Bjorkmann and Eklundh [35] introduced a real-time update of a restricted space of the extrinsic parameters. Pettersson and Petersson [36] extended a robust essential matrix estimation with a fast and robust FPGA-feature extraction. Parameter estimation for every new frame, beginning with rectified images, optimizing the extrinsic rotation and using a Kalman-Filter to limit overfitting was introduced by Hansen et al. [37].

8.3 Extraction of Image Features

Due to its stability and robustness, in respect of the requirements in advanced driver assistance systems, the Scale-Invariant Feature Transform (SIFT) by Lowe [9] is selected for this application as a state-of-the-art image feature descriptor and extractor in order to find sparse pixel correspondences in image pairs of a stereo camera system.

8.3.1 Detection of SIFT-Feature Points

Lowe's SIFT (Scale-Invariant Feature Transform, [9]) is a blob detector, which utilizes Lindeberg's scale-space approach [16] to achieve scale invariance. Blobs are detected by finding local maxima in the approximation of the Laplacian scale-space. The approximation of the Laplace operator is realized by the difference of two low pass filtered images, where both Gaussian kernels consist of different variances. The resulting scale-space approximation, the Difference of Gaussians (DoG), is constructed of several octaves with different image scales (see Figure 8.15). Every octave is subdivided into multiple intervals, which indicate the increasing variance of the Gaussian kernels. The initial interval of each octave arises by subsampling a specific interval of the previous octave. The DoG-pyramid, which represents the edges on multiples scales and different granularities, is browsed for local maxima in three dimensions (image position and intervals). After the detection of feature candidates in the discrete scale-space, their localization is refined by a Taylor series in order to position the candidates with subpixel accuracy and to approximate the extrema in the continuous scale-space.

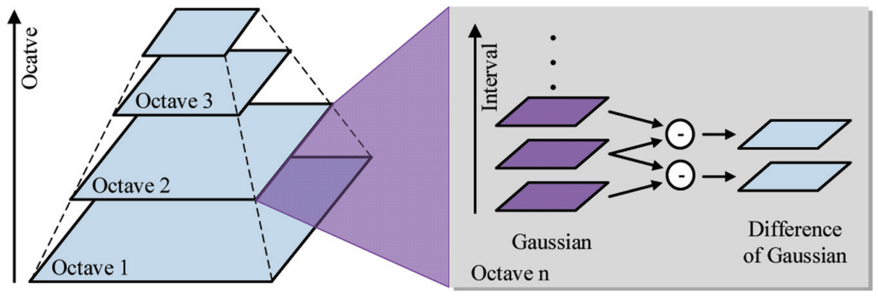


Figure 8.15 Image pyramid. The scale-space is constructed by different octaves, which consists of multiple intervals. Each interval indicates a specific variant of the used Gaussian kernel. In order to approximate the Laplace scale-space, the Difference of Gaussian is determined.

Candidates with a low contrast behavior and too edge like candidates are discarded.

8.3.2 Description of SIFT-Image Features

The SIFT-descriptor is a histogram-based descriptor and provides rotation invariance. Before histogram generation, the main orientation of each image feature is determined in order to align the local image region. To ascertain the main orientation for an image feature, a histogram of local image gradients is generated. The contribution of a local gradient to its corresponding orientation bin is defined by its magnitude and its distance to the feature point. After a smoothing step, the maximal histogram bin represents the main orientation of a feature point.

In addition to a reproducible detection of characteristic image points, a distinctive and robust description of the local neighborhood of the detected points is indispensable. For the description of image features, the gradient magnitude and orientation of the DoG-pyramid is used. A squared pixel area around the detected feature point is rotated by the feature orientation (see Figure 8.12) and subdivided into a grid (see Figure 8.16). For each

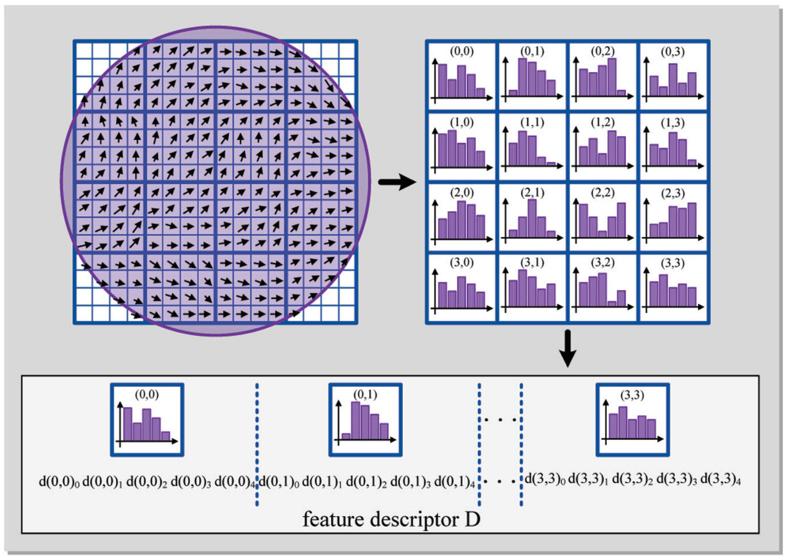

(0,0)
(0,1)
(0,2)
(0,3)
(1,0)
(1,1)
(1,2)
(1,3)
(2,0)
(2,1)
(2,2)
(2,3)
(3,0)
(3,1)
(3,2)
(3,3)
(0,0)
(0,1)
(3,3)
d(0,0)₀ d(0,0)₁ d(0,0)₂ d(0,0)₃ d(0,0)₄ d(0,1)₀ d(0,1)₁ d(0,1)₂ d(0,1)₃ d(0,1)₄ . . . d(3,3)₀ d(3,3)₁ d(3,3)₂ d(3,3)₃ d(3,3)₄
feature descriptor D

Figure 8.16 Generation of feature descriptor. The local neighborhood is subdivided into independent subregions, which are combined into individual histograms. After a weighting and smoothing, the feature descriptor is generated by concatenating the single histograms to as a resulting feature vector.

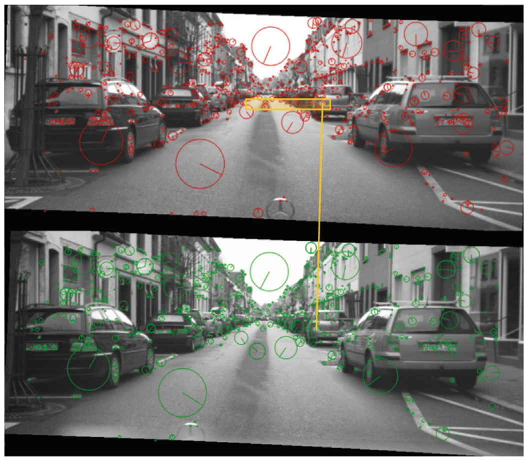


Figure 8.17 Extracted SIFT-features with exemplary geometry-based restriction of matching candidates. By restricting possible matching candidates geometrically, the problem size is significantly reduced.

grid element, an independent histogram of gradients is generated using orientation and magnitudes. The different histograms are weighted, smoothed and combined in a vector, which represents the final feature descriptor. The standard parameters of SIFT, which are suggested by Lowe [9], lead to 128 dimensions with floating point precision for the feature vector.

An exemplary SIFT-feature extraction of a rectified automotive scene is shown in Figure 8.17. The features of the left/right stereo camera are depicted in red/green. The scale of the features is illustrated as the circle's diameter, the orientation of the features with the additional radius line.

8.4 Matching of Image Features

The application of feature matching for advanced driver assistance systems favors correct pixel correspondences instead of a certain set of instable feature matches. Therefore, the matching of image features follows a straight forward approach with a significantly reduced problem size through matching of selected candidates. In this context, it is of minor interest which feature detector and extractor are used for the generation of image features.

Due to the fact, that SIFT is a histogram-based descriptor, a vector norm has to be evaluated as correspondence metric. A trade-off between computational complexity and conclusive results is the sum norm. The matching with sum norm results in marginally lower matching quality compared to matching with the Euclidean norm, but with a localization-based restriction of matching candidates, the matching results yield sufficient accuracy.

By constraining the pool of possible matching candidates, the problem size of feature matching is reduced significantly. The initial brute force matching requires a computation of the correspondence measure between each features of the left image and every feature of the right image. By taking into account the geometric set up of the stereo camera system, the search space is reduced to a fraction of the initial problem size, which results in a noticeable speed-up of matching and less wrong pixel correspondences at the same time (see Figure 8.17).

An exemplary result of the primarily brute force feature matching and for the enhanced matching process using the mentioned algorithmic setup is shown in Figure 8.18. Both stereo input images are overlaid and the image related features are displayed in red/green for the left/right stereo image. The significant increase of matching quality is expressed by the reduction of detected false pixel correspondences (blue connections) in relation to the correct pixel assignments (yellow connections). For the depicted results of feature matching, the sum norm is applied as correspondence measure and a localization-based restriction for choosing matching candidates is used.

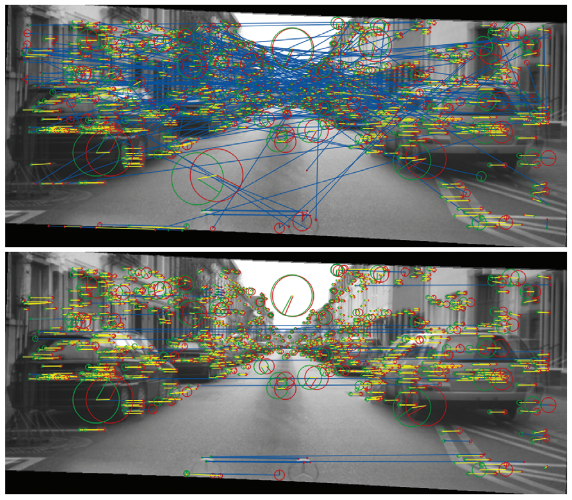


Figure 8.18 Exemplary results of feature matching. The left and right stereo images are overlaid; features of the left/right image are displayed in red/green. Correct matches are depicted in yellow; false matches are shown in blue. The upper image shows the results of the initial brute force matching, whereas the lower image shows the results of the enhanced matching process.

8.5 Extrinsic Online Self-Calibration

Hartley and Zisserman present the fundamentals of extrinsic online self-calibration in their book [38] about multiple view geometry. The extrinsic parameters of a stereo system are described by the rotation $\mathrm{R_X} \in SO(3)$ and the translation vector $\mathrm{t_X} \in \mathbb{R}^3$. Given the extrinsic parameters the transformation of a point $\mathbf{X}_\mathrm{l} \in \mathbb{R}^3$ in the left camera coordinate system into the right camera coordinate system is described as

$$\mathbf{X}_\mathrm{r} = \mathrm{R_X}(\mathbf{X}_\mathrm{l} - \mathrm{t_X}).$$

Normally, extrinsic stereo camera calibration comes down to recovering $\mathrm{R_X}$ and $\mathrm{t_X}$. In the following, $\mathrm{t_X}$ is assumed constant and only $\mathrm{R_X}$ is recovered. During rectification $\mathrm{R_X}$ is broken down into

$$\mathrm{R_X} = \mathrm{R}_r^{-1} R_l$$

in order to determine the rotation of the left and right camera coordinate system to the common image plane respectively.

As decalibration is assumed to vary within a small range of only a few degrees, the recalibration is based on pre-rectified image point correspondences. The images may be pre-rectified using the camera parameters from the initial offline or a previous calibration run.

Given N as the corresponding pre-rectified image points $\widetilde{\mathrm{P}}_i$ and $\widetilde{\mathrm{Q}}_i$ for $i = 1, \ldots, N$ and assuming pinhole camera matrices K for simplicity, the image points are related to their unit directional image vectors

$$\tilde{p}_i \cong \mathrm{K}^{-1}\widetilde{\mathrm{P}}_i$$

$$\tilde{q}_i \cong \mathrm{K}^{-1}\widetilde{\mathrm{Q}}_i.$$

These vectors are related by the common epipolar constraint

$$0 = \widetilde{\mathrm{Q}}_i \mathrm{K}\,\check{\mathrm{R}}\,\mathrm{K}^{-1}\widetilde{\mathrm{P}}_i$$

whereas $\check{\mathrm{R}}$ denotes the rotation compensating the decalibration.

Since the decalibration is assumed to be small, optimization close to the identity matrix has to be avoided due to overfitting. Thus, the image vectors are re-rotated in the original camera coordinate systems via

$$\mathrm{p_i} = \mathrm{R_l}\widetilde{\mathrm{p}}_\mathrm{i}; \quad \mathrm{q_i} = \mathrm{R_r}\widetilde{\mathrm{q}}_\mathrm{i}.$$

Projecting them onto their respective image planes yields

$$\mathrm{P}_i = \mathrm{K}p_i; \quad \mathrm{Q}_i = \mathrm{K}q_i.$$

Given the measured image vector p_i, the depth d_i of the scene point $\mathbf{X}_1$ and the decalibration $\check{\mathrm{R}}$, the corresponding image point Q_i may also be modelled as

$$\mathrm{Q}'_i(\check{\mathrm{R}}, \mathrm{d}_i) = \mathrm{K}\check{\mathrm{R}}\mathrm{R}_\mathrm{X}((p_i\mathrm{d}_i) - \mathrm{t}_\mathrm{X}).$$

Due to noise there is no exact solution, the objective function has to minimize the reprojection error e_i between measured and modelled image points

$$e_i = \|\mathrm{Q}_i - \mathrm{Q}'_i(\check{\mathrm{R}}, \mathrm{d}_i)\|.$$

Thus, the objective function including all image point correspondences is to minimize the sum of all squared reprojection errors and is formulated by

$$\underset{\check{\mathrm{R}},\,\mathbf{d}}{\operatorname{argmin}} \sum_{i=1}^{N} e_i^2$$

with $\mathbf{d} = [\mathrm{d}_1 \ldots \mathrm{d}_N]$. The solution is found by a non-linear optimization method, e.g., Levenberg-Marquardt.

8.6 Application-Specific Algorithmic Parameterization

The manifold varieties of algorithmic parameterizations for feature-based camera self-calibration lead to a sprawling design space, which is barely ascertainable in its entirety. Two exemplary selected application-specific aspects out of this design space are presented in this section. In subsection 8.6.1, the impact of differing bit depth of input images on the extraction of SIFT-features is shown. The parameterization of the presented matching methods is discussed in subsection 8.6.2.

8.6.1 Decreasing Bit Depth of Input Images for Extraction of SIFT-features

The availability of various cameras and the ongoing development of image processor technology lead to stereo systems, which provide digital images with a higher dynamic range. A higher bit depth of 8, 12 or 16 bit per pixel (bpp) promises a higher degree of representable details. However, it is not proven that a feature extractor will extract features of higher quality, when the bit depth for the input images is increased. In case of SIFT-feature extraction for a stereo camera self-calibration, this section shows, that the extracted pixel correspondences for 8 bpp input images and 12 bpp input images lead to identical pixel correspondences.

To ensure full accuracy during computations and to avoid effects of application-specific optimizations, a floating point software version of the SIFT-feature extraction is fed with 8 bpp and 12 bpp input images. Depending on the pixel depth of the input images, a bit depth specific algorithmic parameter set is configured.

After the SIFT-feature extraction, the nearest-neighbor distance ratio matching in combination with a geometry-based restriction of matching candidates (GB NNDR) is applied in order to find corresponding pixels. The experiment is accomplished with a dataset for which rectified input images and related disparity maps exist to validate the detected pixel combinations (see Figure 8.19). By checking the disparity of a match position in the left input image, it is possible to verify the corresponding match position in the right image. A radius offset for the detected matches of $\varepsilon = 0.5$ pixels for the position is tolerated during this investigation. The quantities for the extracted features and detected matches are shown in Table 8.3. The algorithmic parameters for the different SIFT-feature extractions are chosen to yield at least 1,000 features for both input images of the stereo camera system.

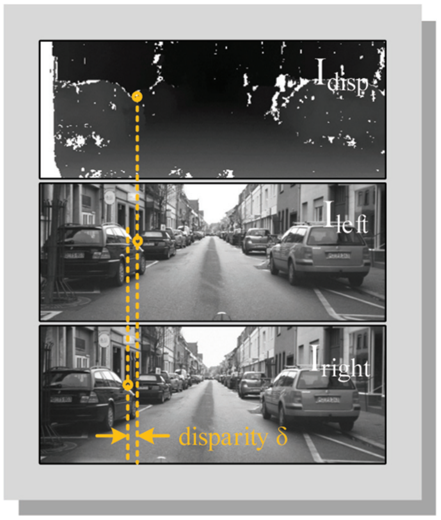


Figure 8.19 Verification of match positions with disparity maps. For rectified images, the horizontal difference of feature positions of a corresponding pixel pair equals the related value of the disparity map. With this technique, it is possible to validate resulting matching lists for datasets with ground truth disparity maps.

Table 8.3 Numbers of extracted SIFT-features and detected matches for 8 bpp input images and 12 bpp images. The number of the geometry-based (GB) nearest-neighbor distance ratio matches (NNDR) drops significantly but ensures a high explicitness of matches. The algorithmic parameters of the SIFT-feature extraction of the two test cases are adjusted in order to extract a similar number of features, which lead to an identical number of verified matches

	8 bpp Image	12 bpp Image
#SIFT-features left image	1,056	1,069
#SIFT-features right image	1,011	1,019
#GB NNB matches	1,013*	1,026*
#GB NNDR matches	608/60.0%	611/59.6%
#disparity verified matches	542/89.1%	544/89.0%
#matches not valid for evaluation	29/4.8%	28/4.9%
#matches wrong correspondences	37/6.1%	39/6.4%

**n* features of the left image have matched with features of the right image; duplicate assignments in the right image possible.

The significant difference between the number of geometry-based NNB matches and geometry-based NNDR matches is caused by the ratio factor, by which equivocal correspondences are rejected. A few correct pixel assignments may be rejected as well using this method, but the matching difference of those pixel pairs is not sufficient small. A valuation of the resulting absolute numbers is beyond the focus of this chapter, but by comparing the differences of the two versions of SIFT-feature extraction and matching it is clear, that there is nearly no difference between using an 8 bpp input image or a 12 bpp input image. To guarantee identical pixel correspondences, a visual inspection of the matching results is mandatory. In Figure 8.20 the result of detected SIFT-features of the left input image (blue: identical matches, orange: exclusive 12 bpp features, red: exclusive 8 bpp features) is shown. Out of 1,069 detected feature positions in the 12 bpp input image, 1,045 (97.8%) identical feature positions are detected again in the 8 bpp input image. In addition, there are 24 (2.2%) exclusive 12 bpp feature positions detected and 14 (1.3%) exclusive 8 bpp feature positions detected. Similar numbers are revealed by comparison for the feature extraction of the different right input images.

After the geometry-based NNDR matching of both feature sets, the comparison of the resulting pairs of the matched pixel correspondences allows a conclusion, if there is a difference between a feature extraction and matching of a 12 bpp input image and a 8 bpp input image. As shown in Figure 8.21, the bulk of the pixel correspondences are identical (blue lines); out of 611 found

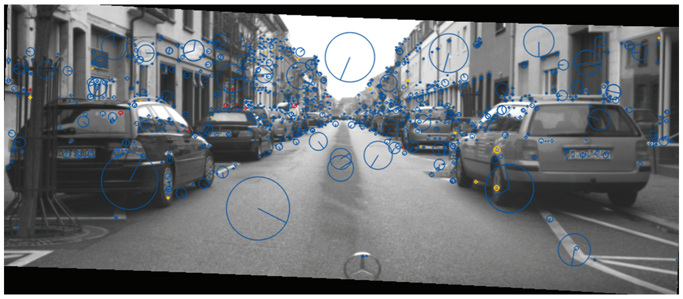


Figure 8.20 Comparison of the resulting SIFT-features of the left input image for 12 bpp images and 8 bpp images. In the 12 bpp input image, an overall number of 1,069 features have been detected, whereas in the 8 bpp input image 1,056 features have been determined. A subset of 1,045 features (97.8%) is identical in both images (blue). There are 14 (1.3%) exclusive 8 bpp feature positions (red) detected and 24 (2.2%) exclusive 12 bpp feature positions (orange).

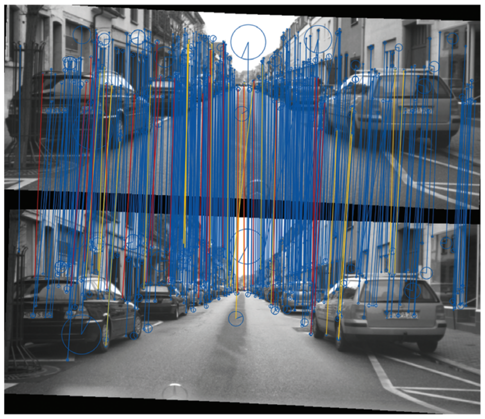


Figure 8.21 Comparison of the resulting pixel correspondences for the 8 bpp and 12 bpp input images. In the 12 bpp input image, an overall number of 611 pixel pairs has been detected, whereas in the 8 bpp input image 608 correspondences have been determined. A subset of 587 pairs (96.1%) is identical in both images (blue lines). Furthermore, there are 23 (3.8%) exclusive 8 bpp pairs (red lines) and 24 (3.9%) exclusive 12 bpp pixel correspondences (orange lines).

correspondences, 587 pairs (96.1%) are equal. In addition, there are 23 (3.8%) exclusive 8 bpp correspondences (red lines) and 24 (3.9%) exclusive 12 bpp correspondences (orange lines).

By tuning the algorithmic parameters in relation to the pixel depth of the used input images in this case study, it is possible to extract identical pixel correspondences. If there is no reason for further image processing steps, which require a proven higher bit depth than an 8 bpp graymap image, it is advisable to process the standard 8 bpp image in order to save computation resources.

8.6.2 Threshold-based Feature Matching

In this context of wide baseline stereo matching, threshold-based feature matching is used. As highlighted in subsection 8.2.2 , a nearest-neighbor-based match is defined as a pair of two descriptors, which are nearest neighbors of a matching process with a descriptor distance below a threshold. Furthermore, a feature only has one matching correspondence. In order to ensure a high rate of correct matches with a low rate of false matches, simultaneously, the threshold has to be selected in accordance to the algorithmic setup and the application-specific image content. Therefore, in this section a method for threshold selection is presented.

Underlying assumption for selecting a threshold for the presented NNB matching is the fact that there are correct matches with a low descriptor distance, false matches with a higher descriptor distance and nothing in between. Again, correct and false matches in this experiment are evaluated with existing disparity maps of the stereo camera system. The descriptor distances of an idealized NNB feature matching is shown in Figure 8.22 (right

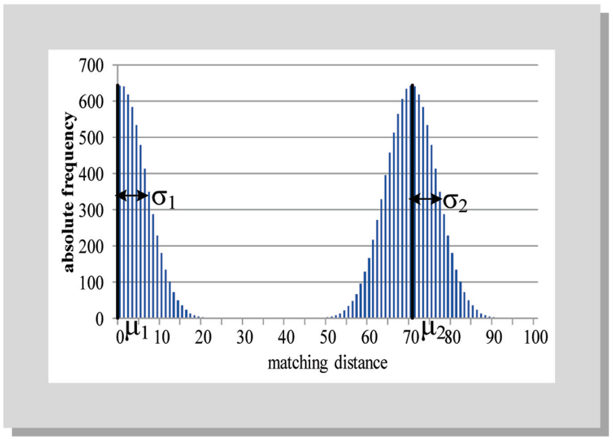


Figure 8.22 Histogram of random generated SIFT-descriptor distances of an idealized NNB feature matching. The right distribution with mean μ_2 displays the distances of wrong matches, whereas the left distribution with mean μ_1 illustrates the correct matches.

plot). For this experiment, 2×10^6 random generated SIFT-descriptors have been generated, pairs have been matched and the distances have been evaluated in a histogram. The resulting distribution of descriptor distances equals the Gaussian distribution, defined by mean μ_2 and deviation σ_2. Obviously, those descriptor distances are false matches. Correct matches follow the same distribution, but with differing mean μ_1 and deviation σ_1, as depicted in Figure 8.22 (left plot). By definition, descriptor distances are sums of absolute values, negative distances are not possible.

By comparing the distance histogram of the synthetic idealized NNB feature matching (see Figure 8.22) with a real-world NNB SIFT-feature matching (see Figure 8.23, left plot), two distinctive differences are noticeable: Firstly, the distance distribution for the correct feature distances and the false feature distance are overlapped and secondly, both distributions are skewed in direction of the others distribution mean value. This distortion is explainable by the fact, that there are always non-avoidable false positives and false negatives during the matching process. Further information concerning the distance distribution is available in [39].

The resulting distance distribution for the NNB SIFT-feature matching is shown in Figure 8.23 (right plot). Based on this plot, a suitable threshold for the matching process has to be extracted. It is desirable to select a threshold, which skips all of the false matches and approves all correct matches, and which corresponds to a threshold between the two ideal distributions. Due to skewing and overlapping of the distributions, there is always a set of false matches, which has to be tolerated by the chosen threshold. Therefore, the

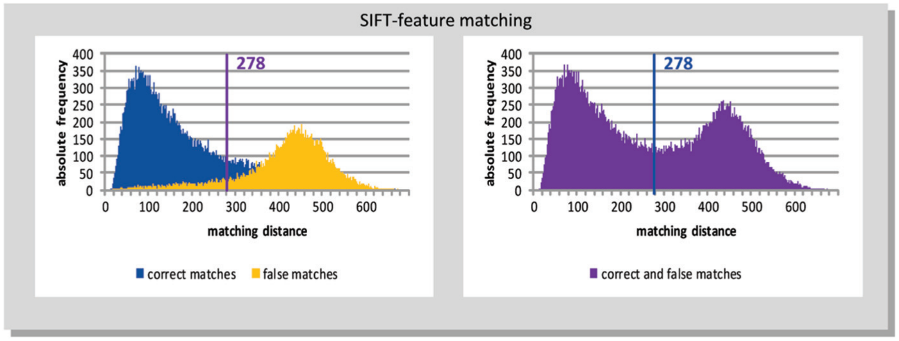


Figure 8.23 Histogram of descriptor distances for a NNB SIFT-feature matching with the extracted threshold according to Otsu. Distances of correct/wrong matches are displayed in blue/orange. The complete distribution is shown in purple.

goal is to minimize the false matches and maximize the correct matches, simultaneously.

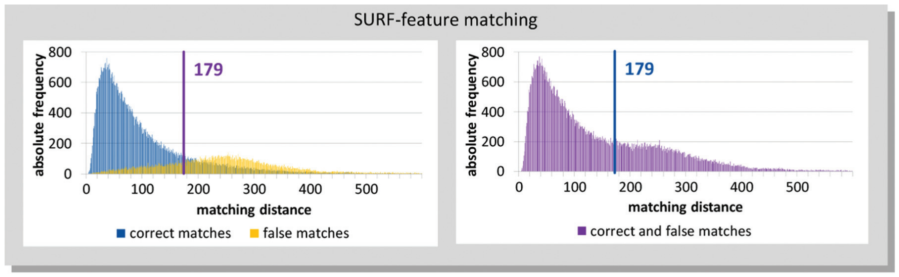
Using the Otsu method [40], two overlapping distributions are separable by applying the discriminant criterion and utilizing the zeroth- and first-order cumulative moments of the distance histogram. Originally, Otsu presented his method for binarization of grey scale images, but the algorithm may be generalized for different types of histogram decomposition. By separating the two Gaussian distributions with Otsu's method, the descriptor distance which divides the distribution into a correct and a false region is determined and set as the matching threshold. Four different case studies have been executed (see Figures 8.23 and 8.24). Even for distance distributions, which do not show such a clear composition of two Gaussian distributions as the SIFT-feature matching case demonstrates, the Otsu's applied method provides reasonable thresholds.

For the entire application of wide baseline stereo matching, the threshold extraction has been performed offline, but it is also conceivable to implement an adaptive frame-to-frame online threshold extraction.

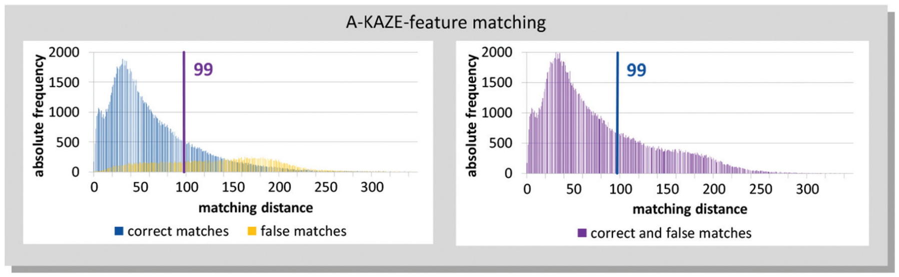
8.6.3 Parameterization of Matching Methods

The aim of this section is the evaluation of the presented matching procedures (see subsection 8.2.2) and the related parameter sets regarding their quality of assigned pixel correspondences in stereo camera systems images. The presented matching methods (TB, NNB, NNDR) result in varying correspondence lists, each of different size and with a variable percentage of correct pixel correspondences. The matching technique, which provides a high rate of correct correspondences for this application and a low rate of wrong assignments simultaneously, has to be identified.

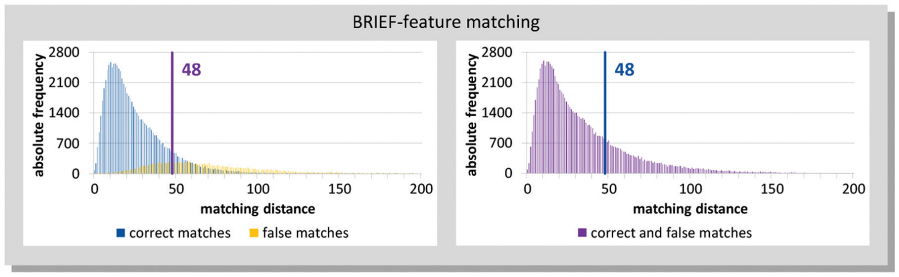
It is possible to speed up the matching process through helpful assumptions about the position of corresponding feature points based on the given geometry of the stereo camera system. Using a spatial pre-selection of detected feature points, the number of candidates for the subsequent descriptor matching is significantly limited. In addition to reducing the problem size for the matching step, the quality of the feature point correspondences is increased. This is caused by excluding matching candidates, which are geometrically contradictory for the used camera setup. Despite the possibility of highly similar descriptors, wrong correspondences are prohibited even before the matching step using this technique.



(a) Case study: SURF-feature matching



(b) Case study: A-KAZE-feature matching



(c) Case study: BRIEF-feature matching

Figure 8.24 Histograms of descriptor distances for different NNB feature matching case studies with the extracted threshold according to Otsu. Distances of correct/wrong matches are displayed in blue/orange. The complete distribution is shown in purple. Due to different descriptors and resulting matching distances, various axis scales for clear presentation are used.

Geometry-based feature matching

The effect of spatial restriction of possible matching candidates (see Figure 8.17) in order to reduce the problem size for the feature matching depends on the permissible window size for matching candidates. In Table 8.4,

Table 8.4 Results for a SIFT-feature matching for a global matching and a geometry-based feature matching. The window size for the geometry-based feature matching is $+/-4$ pixel in y-direction and $+100/-4$ pixel in x-direction

	Global Matching	Geometry-Based Matching
#SIFT-features left image	1,057	1,057
#SIFT-features right image	1,011	1,011
#avg matching candidates	**1,011@1,057 matchings**	**7@1,057 matchings**

an overview of the average number of candidates per matching event is given. In the left/right 8-bit input image, 1,057/1,011 SIFT-features are extracted, which leads to $1{,}011 \times 1{,}057$ descriptor comparisons, when a brute force approach is used. With a window size of $+/-4$ pixel in y-direction (for rectified input images) and $+100/-4$ pixel in x-direction, the average number of descriptor comparisons is reduced to $7 \times 1{,}057$, which is a reduction of problem size of two orders of magnitude. The exact numbers of the candidate distribution for the geometry-based matching are shown in Figure 8.25. The reduction of problem size by a factor of $\times 144$ using the geometry-based feature matching in relation to the global matching clearly outperforms the test, if a detected feature is a matching candidate. Therefore, using the geometry-based matching approach is advisable.

Choosing a matching method

Different methods of feature matching with or without a spatial restriction of the matching candidates directly affect the quality of resulting feature correspondence lists. Exemplary numbers for a variation of matching methods are shown in Table 8.5. Again, in the left/right 8-bit input image, 1,057/1,011

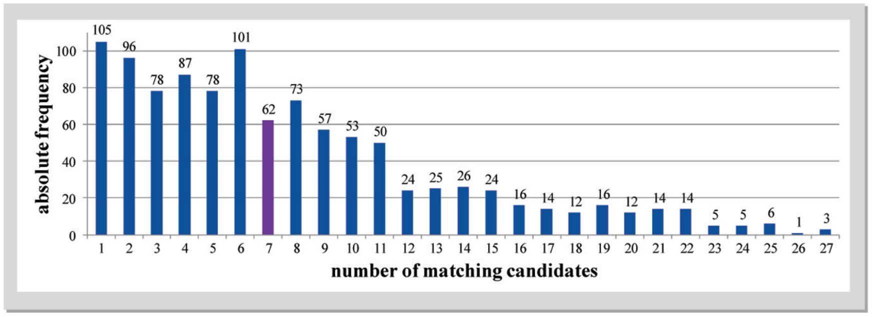


Figure 8.25 Exemplary histogram for the distribution of matching candidates for the geometry-based feature matching (see Table 8.4). The average number of candidates is 7 candidates per matching event.

Table 8.5 Results of disparity verified feature correspondences for different combinations of global and spatial restriction matching methods. In addition to a high rate of correct matches, a minimal number of pixel correspondences has to be given for a reliable subsequent image processing. The total numbers of detected matches for selected algorithmic combinations are given in brackets. The number of correct matches and wrong matches do not result in 100% because of missing values in the ground truth disparity maps. Those values are skipped for evaluation

	Global Matching		Geometry-Based Matching	
	#Correct Matches	#Wrong Matches	#Correct Matches	#Wrong Matches
#TB disparity verified matches	562 (1,057)	400	702 (1,006)	240
	53.2%	37.8%	69.8%	23.9%
#NNB disparity verified matches	541 (735)	149	**556 (597)**	**22**
	73.6%	20.3%	**93.1%**	**3.7%**
#NNDR disparity	493 (540)	31	542 (605)	36
verified matches	91.3%	5.7%	89.6%	6.0%

SIFT-features are extracted. The total number of detected matches for each algorithmic combination is given in brackets (see Table 8.5).

For each method, the geometry-based feature matching grants an improvement of the correct matching rate or the rate remains in the same order of magnitude. The resulting correspondence lists generated with the threshold-based feature matching has the highest number of entries, but the quota of correct matches is insufficiently low. A combination of NNB-matching and the geometry-based restriction leads to the highest rate of correct matches (93.1%) and a low rate of wrong matches (3.7%), simultaneously. Furthermore, the absolute number of correct matches (556) guarantees a stable base for following image processing algorithms. Therefore, the use of the NNB-matching with a geometry-based restriction of matching candidates in order to extract pixel correspondence lists for a feature-based camera self-calibration is recommended.

Accuracy of localization

All prior investigations in this section are based on the assumption that 'disparity verified matching' defines the consensus of the extracted feature-based disparity including a small offset ε and the related actual disparity taken from the disparity ground truth map. This offset ε is necessary in order to tolerate small deviations of feature positions, which are caused during the localization step.

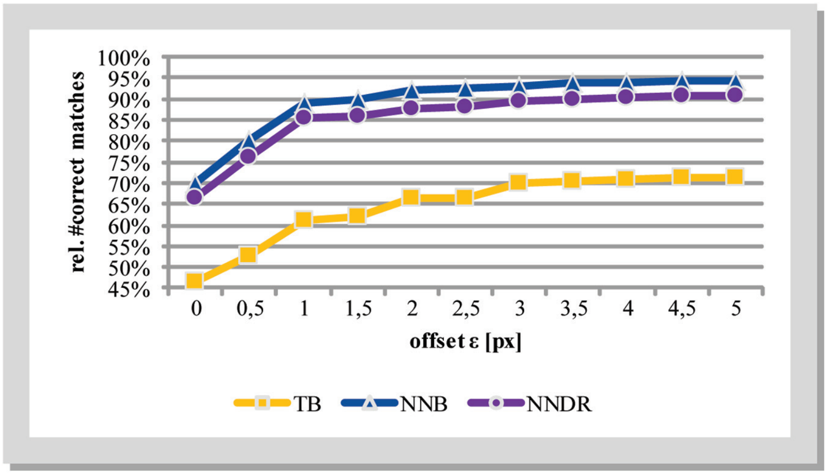


Figure 8.26 Rates of disparity verified pixel correspondences for different offsets ε and three matching methods. For all methods, the rate of correct matches runs into saturation. The NNB matching method performs best over all offsets ε. (TB: Threshold-Based Matching; NNB: Nearest-Neighbor-Based Matching; NNDR: Nearest-Neighbor Distance Ratio Matching).

To evaluate the impact of varying offsets ε, in the left/right 8-bit input image, 1,057/1,011 SIFT-features are extracted and matched with a geometry-based approach for the TB, NNB and NNDR matching method. The rates of disparity verified pixel correspondences for different offsets ε are shown in Figure 8.26. Remarkably the qualitative trend is identical for all matching methods. Furthermore, all methods run into saturation for offsets higher than 3 pixel. As expected, the threshold-based matching (TB) provides the lowest matching rate for all offsets ε. The nearest-neighbor based (NNB) matching method results constantly in the highest rate for disparity verified matches with approximately over 90% (<537 out of 597 matches) for an offset larger than 1 pixel. It is worth mentioning that 70% of all matches (419 out of 597 matches) for the NNB method are identical to the ground truth disparity map (offsets $\varepsilon = 0$ pixel).

To achieve an applicable trade-off between exact 'disparity verified correspondences' and permitting localization errors due to viewpoint changes, all prior investigations have been verified with an offset $\varepsilon = 3$ pixel.

8.7 Hardware Based SIFT-Feature Extraction

Fast and reliable extraction of SIFT-features in the presented context of feature-based camera self-calibration requires a tuned implementation of the algorithm for the hardware platform used. Therefore, in this section, the relevant hardware properties of SIFT-feature extraction are introduced and an overview of existing SIFT-feature implementations is given.

8.7.1 Challenges of SIFT-Feature Extraction

The extraction of SIFT-features is a challenging task due to the number of operations and memory accesses that have to be executed. As depicted in Figure 8.27, the algorithmic steps of SIFT-feature extraction differ in varying ratios of control complexity and regular arithmetic. As shown in [41], the building of scale-space, which consists of multiple separable and symmetric Gaussian filters, is an arithmetically intensive task with almost no control overhead. In contrast, parts of the feature points detection or the descriptor generation require control mechanisms, which result in heavy branching on conventional processors. Furthermore, the scale-space is mandatory for the feature description and has to be buffered until the generation of descriptors, which requires a large memory and arbitrarily non-aligned memory accesses aggravate the challenging memory bottleneck. In addition, the algorithmic quality of SIFT has to be ensured for subsequent processing steps, which requires an appropriate level of internal accuracy of the temporal results.

Therefore, specialized architectures are necessary to ensure the processing performance demanded for SIFT-feature extraction. At the same time, those specialized systems have to be as flexible as possible to guarantee a fast

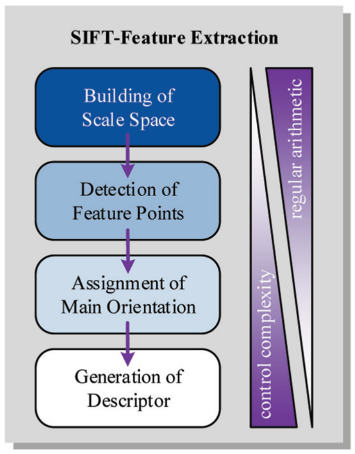


Figure 8.27 Break down of SIFT-feature extraction into four algorithmic steps and relating qualitatively quota of control complexity and complexity (i.e., regular arithmetic).

implementation of future algorithms which might perform better compared to state-of-the-art feature extractors [42].

8.7.2 Existing Systems for Hardware Based SIFT-Feature Extraction

In the following Table 8.6, a set of existing systems/platforms for the hardware based SIFT-feature extraction is presented. The selection shown is not meant to be exhaustive, but elucidates the trade-off of different platforms regarding sufficient processing power, low power consumption and satisfactory flexibility for future algorithm implementations.

Moren et al. [43] presented in 2015 a comprehensive survey of a SIFT-feature extraction for homogeneous and heterogeneous CPU/GPU systems. With different techniques for parallelization and a portable performance concept using OpenCL (Open Computing Language), the SIFT-feature extraction has been implemented on various single device and multi-device platforms.

Table 8.6 Overview of existing systems for SIFT-feature extraction

	Implementation				Frequency	Performance
	Author	Year	Device	Powre	(MHz)**	(fps)
CPU & GPU	Moren et al. [43]	2015	Nvidia GTX 780 TI	250 W*	875	137.6 @ 640x480
			AMD R9-290	300 W*	947	98.7 @ 640x480
			Nvidia GTX 580	244 W*	772	77.2 @ 640x480
			Nvidia Tesla C2050	238 W*	1150	74.0 @ 640x480
			Intel MIC 3120A	300 W*	1100	16.8 @ 640x480
			Intel Core-i7 4930K	130 W*	3400	32.6 @ 640x480
			Intel Xeon E5-2667	130 W*	2900	28.3 @ 640x480
			AMD Opteron 6168	115 W*	1900	8.0 @ 640x480
			Intel Xeon E5-2667	130 W*	2900	4.0 @ 640x480
Mobile GPU	Rister et al. [44]	2013	Snapdragon S4	$\sim$4 W	1,700/400	9.9 @ 320x240
			Nexus 7	N/A	1,600/520	8.6 @ 320x240
			Galaxy Note II	N/A	1,600/400	7.6 @ 320x240
			Tegra 250	$\sim$3 W	1,000/333	7.9 @ 320x240
FPGA & ASIC	Bonato et al. [45]	2008	Altera Stratix II	N/A	100	30.0 @ 320x240
	Yao et al. [46]	2009	Xilinx Virtex 5	N/A	100	32.3 @ 640x480
	Huang et al. [47]	2012	TSMC 18μm CMOS	N/A	100	30.0 @ 640x480
	Yum et al. [48]	2015	Xilinx Virtex 6	N/A	170	36.9 @ 1280x720
ASIP	Mentzer et al. [41, 42]	2015	TSMC 45nm process	<1 W	400	1 @ 800x640

*Thermal Design Power.
**For category mobile GPU: CPU/GPU frequency.

The systems are separated into four different implementations, where each implementation is optimized according to device specific characteristics:

- Host-device implementation for control
- GPU device implementation
- Multi-core CPU device
- Multi-device implementation

The systems are evaluated for multiple image sizes for equal algorithmic setups. Single device runtimes are listed in Table 8.6 for VGA image size. Noticeable is the fact, that all single GPU systems and multi-device systems, in which a GPU is enlisted, provide enough performance for a real-time SIFT-feature extraction for VGA images, but require more than 230 W power consumption. Furthermore, CPU single device systems are close to real-time by providing 17–32 fps, but again, the power consumption is far too high for use in automobiles with over 115 W power consumption. The AMD Opteron 6168 and Intel Xeon E5 do not reach a sufficient frame rate for a SIFT-feature extraction application. The author presents three different heterogeneous systems, which are assembled by the afore mentioned single device systems, which provide enough performance for real-time applications even for very large images. For all systems, the flexibility is ensured by using the high-level OpenCL.

In 2013, Rister [44] proposed an investigation of SIFT-feature extraction on four different platforms using mobile GPUs. The author used a heterogeneous dataflow scheme and applied a partitioning of workload between CPU and GPU. Different platform specific optimizations are used, e.g., data compressing by pixel reordering or branchless convolution through on-the-fly code generation. With frame rates reaching between 7.6 fps and 9.9 fps, the performance is too poor for an use in ADAS, but a power consumption of the complete systems of <5 W fulfills the requirements demanded. Furthermore, flexibility is guaranteed by OpenGL for Android.

Bonato et al. presented 2008 the first hardware based SIFT implementation [45]. The heterogeneous system consists of a hardware accelerator for SIFT-feature detection and a NIOS II softcore processor for SIFT-descriptor generation. The system has been emulated on an Altera Stratix II FPGA and a frame rate of 30 fps for QVGA images has been reached.

One year later in 2009, Yao et al. claimed to reach a comparable frame rate of 32.3 fps, but for VGA images. They presented a hardware-based SIFT-feature detector, which has been emulated on a ML507 board, and a SIFT-feature generation in software. The drawback of the presented work is the simplified SIFT scale-space, which leads to a limited algorithmic quality, compared to the original algorithm.

The first fully hardware-based SIFT-feature extraction has been presented in 2012 by Huang et al. [47]. The author's system reaches a frame rate of 30 fps for VGA images and uses a TSMC 180 µm CMOS process.

In 2015, Yum et al. proposed a FPGA-based full SIFT implementation, which is capable of processing 36.85 fps for HD images on a Xilinx Virtex 6 device [48]. By reducing the amount of necessary internal memory and a local-patch reuse scheme, a high data throughput is reached, but the building of scale-space is adjusted, which affects the algorithmic quality.

These hardware-based approaches provide adequate processing power for a high frame rate and a sufficiently low power consumption of typically $<10\,\mathrm{W}$, but the presented systems are not SW-flexible.

Mentzer et al. [41, 42] presented an ASIP-based SIFT-feature extraction, which preserves the full algorithmic quality. Sufficient flexibility for future algorithms of image feature extraction is ensured by the platform-specific attribute of full software programmability. The drawback of the presented case study is the low frame rate in FPGA emulation, which prohibits a real time application in automotive use.

Thus, heterogeneous systems consisting of dedicated hardware for accelerating the scale-space construction and a processor-based descriptor generation is a promising trade-off between flexibility, performance and power consumption. State-of-the-art conventional CPUs and GPUs are too power greedy, nowadays mobile GPUs do not reach sufficient frame rates and pure hardware-based systems do not fulfill the requirements for flexibility. A trade-off concerning flexibility by supporting a processor with non-programmable hardware accelerators is a possible approach for a SIFT-feature extraction in the field of Advanced Driver Assistance Systems.

8.8 Conclusion

In this chapter, selected aspects of self-calibration for wide baseline stereo camera systems for automotive applications have been introduced. Starting at the extraction and matching of image features up to the extrinsic online self calibration of stereo camera systems, fundamental algorithms have been presented. A promising algorithmic combination consisting of the extraction of SIFT-features, nearest-neighbor-based matching with spatial selection of matching candidates and the estimation of camera parameters in order to rectify misaligned stereo images have been discussed in detail.

Three exemplary aspects of algorithmic parameterizations, which are the impact of a decreasing bit depth of input images, the selection of a matching method and the threshold selection for the matching process, have been examined in detail to show substitutionally the complexity of adjusting existing algorithms to new applications.

In the last section, basic challenges of hardware-based SIFT-feature extraction are presented and hardware-specific solutions for the afore mentioned algorithmic challenges are discussed. Finally, existing systems for the extraction of SIFT-features are reviewed.

As discussed in this chapter, there is no state-of-the-art hardware implementation for the proposed algorithmic combination, which fulfills the three requirements for ADAS, and delivers sufficient processing performance, low power consumption and full flexibility for future algorithms. Thus, remaining challenges will be solved to improve safety for vulnerable road users and to enhance comfort in future automobiles.

References

[1] K. Genuit, Sound-Engineering im Automobilbereich, Springer, 2010.

[2] C. Schmid, R. Mohr and C. Bauckhage, "Evaluation of Interest Point Detectors," *International Journal of Computer Vision*, pp. 151–172, 2010.

[3] K. Mikolajczyk and C. Schmid, "A Performance Evaluation of Local Descriptors," *IEEE Transcations on Pattern Analysis and Machine Intelligence,* pp. 1615–1630, 2005.

[4] H. Aanæs, A. L. Dahl and K. S. Pedersen, "Interesting Interest Points," *International Journal of Computer Vision,* pp. 18–35, 2012.

[5] S. Gauglitz, T. Höllerer and M. Turk, "Evaluation of Interest Point Detectors and Feature Descriptors for Visual Tracking," *International Journal of Computer Vision,* 2011.

[6] J. Heinly, E. Dunn and J.-M. Frahm, "Comparative Evaluation of Binary Features," *ECCV,* pp. 759–773, 2012.

[7] K. Mikolajczyk and C. Schmid, "Scale & Affine Invariant Interest Point Detectors," *International Journal of Computer Vision,* pp. 63–86, 2004.

[8] J. Shi and C. Tomasi, "Good Features to Track," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 593–600, 1994.

[9] D. G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the International Conference on Computer Vision,* pp. 1150–1157, 1999.

[10] H. Hirschmüller, "Accurate and efficient stereo processing by semi-global matching and mutual information," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[11] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1330–1334, 2000.

[12] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. V. Gool, "A Comparison of Affine Region Detectors," *International Journal of Computer Vision,* pp. 43–72, 2005.

[13] J. Canny, "A Computational Approach to Edge Detection," *Transactions on Pattern Analysis and Machine Intelligence,* 1986.

[14] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Proceedings of the Alvey Vision Conference,* pp. 23.1–23.6, 1988.

[15] E. Rosten and T. Drummond, "Machine Learning for high-speed corner detection," *European Conference on Computer Vision,* 2006.

[16] T. Lindeberg, *Scale-Space Theory in Computer Vision*, Kluwer Academic Publishers, 1994.

[17] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," *Proceedings of Conference on Computer Vision and Pattern Recognition,* 2004.

[18] S. Belongie, J. Malik and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *Transactions on Pattern Analysis and Machin Intelligence,* pp. 509–522, April 2002.

[19] E. Tola, V. Lepetit and P. Fua, "DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo," *Transactions on Pattern Analysis and Machine Intelligence,* pp. 815–830, May 2010.

[20] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha and P. Fua, "BRIEF: Computing a Local Binary Descriptor Very Fast," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1281–1298, 2012.

[21] A. Alahi, R. Ortiz and P. Vandergheynst, "FREAK: Fast Retina Keypoint," *Conference on Computer Vision and Pattern Recognition,* pp. 510–517, 2012.

[22] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," *International Conference on Computer Vision,* pp. 2564–2571, 2011.

[23] S. Leutenegger, M. Chli and R. Y. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," *International Conference on Computer Vision,* pp. 2548–2555, 2011.

[24] P. F. Alcantarilla, A. Bartoli and A. J. Davison, “KAZE Features,” *Proceedings of the 12th European Conference on Computer Vision,* pp. 214–227, 2012.

[25] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, “SURF: Speeded Up Robust Features,” *Journal of Computer Vision and Image Understanding,* pp. 346–359, 6 2008.

[26] P. Alcantarilla, J. Nuevo and A. Bartoli, “Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces,” *Proceedings of the British Machine Vision Conference,* 2013.

[27] M. Muja and D. G. Lowe, “Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration,” *International Conference on Computer Vision Theory and Applications,* pp. 331–340, 2009.

[28] H. C. Longuet-Higgins, “A Computer Algorithm for Reconstructing a Scene from Two Projections,” *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms,* pp. 61–62, 1987.

[29] R. I. Hartley, “Estimation of relative camera positions for uncalibrated cameras,” *European Conference on Computer Vision,* pp. 579–587, 1992.

[30] Z. Zhang, Q.-T. Luong and O. Faugeras, “Motion of an Uncalibrated Stereo Rig: Self-calibration and Metric Reconstruction,” *IEEE Transactions on Robotics and Automation,* pp. 103–113, 1996.

[31] Q.-T. Luong and O. D. Faugeras, “Self-Calibration of a Moving Camera from Point Correspondences and Fundamental Matrices,” *International Journal for Computer Vision,* pp. 261–289, 1997.

[32] P. Torr and A. Zisserman, “Robust Computation and Parametrization of Multiple View Relations,” in *Computer Vision and Image Understanding*, 2000.

[33] B. Triggs, P. F. McLauchlan, R. I. Hartley and A. W. Fitzgibbon, “Bundle Adjustment – A Modern Synthesis,” *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice,* pp. 298–372, 2000.

[34] T. Dang, C. Hoffmann and C. Stiller, “Continuous Stereo Self-calibration by Camera Parameter Tracking,” *Transactions on Image Processing*, pp. 1536–1550, 2009.

[35] M. Björkman and J.-O. Eklundh, “Real-Time Epipolar Geometry Estimation of Binocular Stereo Heads,” *Transactions on Pattern Analysis and Machine Intelligence,* pp. 425–432, 2002.

[36] Petterson and Petterson, “Online stereo calibration using FPGAs,” *Intelligent Vehicles Symposium,* 2005.

[37] P. Hansen, H. S. Alismail, P. Rander and B. Browning, "Online Continuous Stereo Extrinsic Parameter Estimation," *International Conference on Computer Vision and Pattern Recognition,* 2012.

[38] R. I. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2004.

[39] R. Szeliski, Computer Vision: Algorithms and Applications, London: Springer-Verlag, 2011.

[40] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man and Cybernetics,* pp. 62–66, 1979.

[41] N. Mentzer, G. P. Vaya and H. Blume, "Analyzing the Performance-Hardware Trade-off of an ASIP-based SIFT Feature Extraction," *Journal of Signal Processing Systems,* 2015.

[42] N. Mentzer, N. V. Egloffstein, G. P. Vaya, W. Ritter and H. Blume, "Instruction-Set Extension for an ASIP-based SIFT Feature Extraction," in *International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS)*, Samos, Greece, 2014.

[43] K. Moren and D. Göhringer, "A framework for accelerating local feature extraction with OpenCL on multi-core CPUs and co-processors," *Journal of Real-Time Image Processing,* pp. 1–18, 03 2016.

[44] B. Rister, G. Wang, M. Wu and J. R. Cavallaro, "A Fast and Efficient SIFT Detector using the mobile GPU," *IEEE International Conference on Acoustics, Speech and Signal Processing,* 2013.

[45] V. Bonato, E. Marques and G. A. Constantinides, "A Parallel Hardware Architecture for Scale and Rotation Invariant Feature Detection," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1703–1712, 2008.

[46] L. Yao, H. Feng, Y. Zhu, Z. Jiang, D. Zhao and W. Feng, "An architecture of optimised SIFT feature detection for an FPGA implementation of an image matcher," *International Conference on Field-Programmable Technology,* pp. 30–37, 2009.

[47] F.-C. Huang, S. Huang, J. Ker and Y. Chen, "High-Performance SIFT Hardware Accelerator for Real-Time Image Feature Extraction," *IEEE Transactions on Circuits and Systems for Video Technology,* pp. 340–351, 03 2012.

[48] J. Yum, C.-H. Lee, J.-S. Kim and H.-J. Lee, "A Novel Hardware Architecture with Reduced Internal Memory for Real-time Extraction of SIFT in an HD Video," *IEEE Transactions on Circuits and Systems for Video Technology,* 2015.