7 Application-oriented Assessment of Computer Vision Algorithms

Peter Klausmann[1], Stefan Fries[1], Dieter Willersinn[1], Uwe Stilla[2], and Ulrich Thönnessen[2]

[1]Fraunhofer-Institut für Informations- und Datenverarbeitung (IITB) Karlsruhe, Germany
[2]Forschungsinstitut für Informationsverarbeitung und Mustererkennung Ettlingen, Germany

7.1 Introduction 133
7.2 Analytical versus empirical performance analysis 134
7.2.1 Analytical performance characterization 134
7.2.2 Empirical performance characterization 135
7.3 Application-oriented empirical algorithm assessment 136
7.3.1 Task, performance criteria and requirement profile . 137
7.3.2 Assessment data 138
7.3.3 Image data acquisition 140
7.3.4 Provision of training data for development 140
7.3.5 Algorithm performance characterization 140
7.3.6 The assessment function 141
7.3.7 Overview of the assessment procedure 142
7.4 The assessment system 143
7.5 Example: Assessment of a detection algorithm 145
7.6 Conclusion 149
7.7 References 149

7.1 Introduction

The crucial question to be answered during the assessment of a computer vision algorithm is to what extent is the algorithm performance useful? The utility of an algorithm can only be stated with respect to an application, hence the assessment of computer vision algorithms

Handbook of Computer Vision and Applications
Volume 3
Systems and Applications

ISBN 0-12-379773-X/$30.00

is only possible if the application of the computer vision algorithm is given.

The assessment of a computer vision algorithm falls into two parts, namely, algorithm performance characterization, and the decision as to whether the attained performance is sufficient for the given application. For an application-oriented assessment of computer vision algorithms both questions must be addressed.

Section 7.2 presents an overview of approaches to characterize the performance of computer vision algorithms. Section 7.3 describes our concept for the application-oriented assessment of computer vision algorithms. The concept covers both the performance characterization and the assessment of algorithm performance. Our assessment system is described in Section 7.4. Section 7.5 describes the assessment of a detection algorithm for two different applications.

7.2 Analytical versus empirical performance analysis

Models to explain and to predict the performance of a computer vision system have gained increasing attention among the computer vision community during the last few years [1, 2, 3, 4, 5, 6]. Although there exist a considerable number of approaches in computer vision, the design of systems that are able to work accurately over a broad range of images is still difficult. Two mainstream approaches have been developed by researchers to characterize the performance of image processing algorithms. The first approach has concentrated on analytical investigation and sophisticated concepts of error propagation. The second approach has focused on the quantitative, empirical and experimental performance characterization.

7.2.1 Analytical performance characterization

In nearly all disciplines of image processing ideas have been proposed concerning algorithmic behavior under perturbation influence. In linear estimation a complete theory about error propagation in terms of covariance calculus exists [7]. Haralick [8] and Bar-Shalom and Li [9] generalized such approaches to nonlinear problems. Also, numerous analytical approaches have been proposed to investigate edge detectors [10, 11, 12, 13], which are a general front-end tool of image analysis systems. Texture recognition and image segmentation approaches have been evaluated analytically since the beginning of the 1980s [14, 15, 16].

Another major topic addressed by researchers is the control of random perturbations and the influence of clutter. They have attempted to relate the results of a vision system to image characteristics and to investigate the main influence factors on algorithm performance.

There are many reasons why the image processing methodology may fail [17, 18]: misinterpretations of data features caused by noise; inconsistent description of object features; missing contrast of the objects; imprecise edge detection; or background clutter. Therefore, the object hypotheses usually are not consistent with all data, not even with a subset for which an algorithm is trained. Thus, the characteristics of objects to be detected have been analyzed under different noise and clutter conditions and have been compared with the capacity of human observers. Several papers discuss the features of objects and background concerning the visibility to human observers. Conners and Ng [19] modeled the human preattentive vision system on the basis of gray-value co-occurrence matrices. Shirvaikar and Trivedi [20] used these matrices to describe the background clutter with a texture-based image clutter (TIC) measure and discussed the influence of this clutter measure on the detection performance and false alarm rates. Waldman et al. [21] suggested that spatial extent, contrast, clutter, movement, shape, number of targets and color play the main roles in performance prediction. Schmieder and Weathersby [22] first noticed the relevance of the relation between target and background in determining detection probability. They defined a signal-to-clutter ratio (SCR) and compared the detection capacity of human observers versus the SCR and several target resolutions. Ratches et al. [23] stated that the signal-to-noise-ratio (SNR) defined by the average contrast difference between target and background normalized by the average background intensity has a basic influence on the detection performance.

The main problem with analytical descriptions is that most vision systems consist of a combination of multitudinous steps. Each layer includes highly adapted parameters for a specific application task. The complexity of the computational procedure can make a complete analytical analysis infeasible. Additionally, both the applications as well as the different methodologies are hardly comparable and the disadvantage of one method is the main strength of another. Heuristic parts of the algorithm are specifically adapted to special situations and are seldom valid in a more general context. For this reason empirical performance characterization is to date a common way to answer the earlier-stated question, although an analytical answer would be highly desirable. Suggestions of how to treat the evaluation problem are discussed in the following.

7.2.2 Empirical performance characterization

An early comparative study of image processing algorithms is that reported by Weszka et al. [24]. They analyzed three feature-based texture-measure approaches and evaluated their performance using a fixed data set. Nowadays, it is still a common method to consider differ-

ent sets of input images and to relate the performance of the algorithm to the variation of image and algorithm parameters. This topic is addressed by Kanungo et al. [25]. They started with a set of noise-free images and generated a large number of images from this base set by adding different realizations of noise and perturbations. They then developed operating characteristic curves of the algorithm errors as functions of signal-to-noise ratios and studied the perturbation influence on the algorithm performance. Haralick [26] provided some reference operating curves of false alarm rates versus the nondetection rate as a function of the number of samples and a prespecified error rate of the vision system. To formulate the evaluation problem in terms of empirical statistics, similar approaches have been employed by Bradley [27], Haralick [28], Liu et al. [29], Takeshita and Toriwaki [30], and Nyssen [31]. They reviewed the theory about receiver operating characteristics, hypothesis testing procedures and significance measures, providing additional background for statistical software validation. To treat procedures of considerable computational complexity modern resampling methods such as bootstrap have been used by Cho et al. [32], Jain et al. [33] and Chernick et al. [34]. To evaluate complex algorithms, Courtney et al. [35] suggested that each layer be modeled separately as probability density functions and then combined to predict detection and nondetection rates.

7.3 Application-oriented empirical algorithm assessment

The assessment concept presented in this section eventually yields a qualitative or quantitative statement of algorithm utility for a given application, or a ranking of several algorithms based on that statement of utility. The concept clearly distinguishes between the measurement of algorithm performance and the assessment of measured algorithm performance for a given application. Due to the lack of analytical models when predicting algorithm performance, care must be taken during the acquisition of test data in order to obtain relevant data that reflect the particular difficulties encountered during the prospective application. Another consequence of the stated lack of analytical models is that relevant data have to be provided to the developers so that they can adapt their algorithms to the particular difficulties of the application.

The subsequent assessment of the measured performance is based on a description of the requirements of the application, the *requirement profile*. The algorithm performance is compared to the requirement profile by means of the assessment function that is used to derive an application-specific *figure of merit* (FOM). This FOM can be used in two ways: first, to derive a statement as to whether or not the assessed algorithm is useful for the prospective application; and second, it may be

used for ranking several algorithms according to their figure of merit. The last step of the assessment concept is to collect all of the information produced during the measurement and assessment procedure (e.g., performance profile, the assessment function, the analysis of single algorithm results, etc.) in an assessment profile that is presented to the prospective algorithm user as well as to the developer.

7.3.1 Task, performance criteria and requirement profile

A computer vision algorithm always performs its task in the context of an application [36]. Hence, the assessment of a computer vision algorithm can only be done with respect to the application. Further, it requires an agreement based on general criteria of engineering. This agreement must contain an explicit description of the algorithm task and has to define:

- parameters of the scene;
- objects of interest and their parameters;
- sensor type and sensor parameters;
- the expected computer vision results; and
- the prospective use of the computer vision results.

From the task definition *performance criteria* are derived. We distinguish between general criteria and criteria that are specific to the algorithm class. Examples for algorithm classes are detection, classification or parameter estimation. Typical performance measures for detection algorithms are, for example, detection rate and the false alarm rate. The performance of classification algorithms is usually described by confusion matrices, and parameter estimation algorithms can be characterized by estimation errors and standard deviations.

General criteria important for the user could be availability, reliability, sensitivity with respect to parameter variation, hardware requirements of the algorithm as well as resource consumption and execution time.

The intended use of the computer vision algorithm determines which criteria are of importance, and allows assigning required values to each of the selected criteria. Also, weighting factors (costs) for the selected criteria according to their importance for the application could be established.

The forementioned criteria and weights are fixed in the *requirement profile*. Together with a rule for combining the weighted criteria to obtain the figure of merit they define the so-called *assessment function*. The investigation of additional algorithm features (hardware requirements, etc.) would be described in the requirement profile as well.

7.3.2 Assessment data

This section describes *assessment data*, that is, all data that are necessary to perform the assessment of a computer vision algorithm: image data; collateral data; and truth data.

Type of image data. In order to verify an algorithm we distinguish between: (i) synthetic image data; (ii) image data of modeled objects; and (iii) image data of real objects [37].

Synthetic image data can be computed from scene descriptions by algorithms for computer graphics and virtual reality. There are many tools available to produce images by using a sensor model, an illumination model and an object or scene model. The advantage of using synthetic objects is the inherent knowledge about the truth of the modeled scene. This allows for a testbed structure for fully automatic testing [38].

With the growing demand for the photorealistic appearance of synthetic objects there has been a considerable increase in efforts to model and generate images. For some examinations image data of *model objects* offer an alternative [39]. The image can be taken regardless of the weather, at low cost and under reproducible conditions.

Working under changing conditions or using outdoor scenes the vision system has to cope with data that contain distortions and other perturbations from various sources. To date, it is not possible to realistically model all sources of perturbations that have an impact on the imaging process. For this reason the use of image data of *real objects* is a must for the assessment of a computer vision algorithm.

Collateral data. Besides the information given by the test images supplementary information may be available to the algorithm during operation. Such information includes geometric resolution, radiometric resolution, sensor geometry, and operational conditions. This information is called *collateral data*.

Truth data. The previous sections described the data available to the algorithm. For assessment purposes, there has to be a description of the scene represented by the image data as well. Figure 7.1 illustrates the data acquisition for the assessment of an algorithm that analyzes images of natural scenes. The scene description is referred to as *ground truth*.

It is important to mention here that the ground truth is application-dependent in the sense that it must contain that information that is relevant for the given application. In the case illustrated by Fig. 7.1, the relevant information consists of descriptions of the vehicles present in the scene.

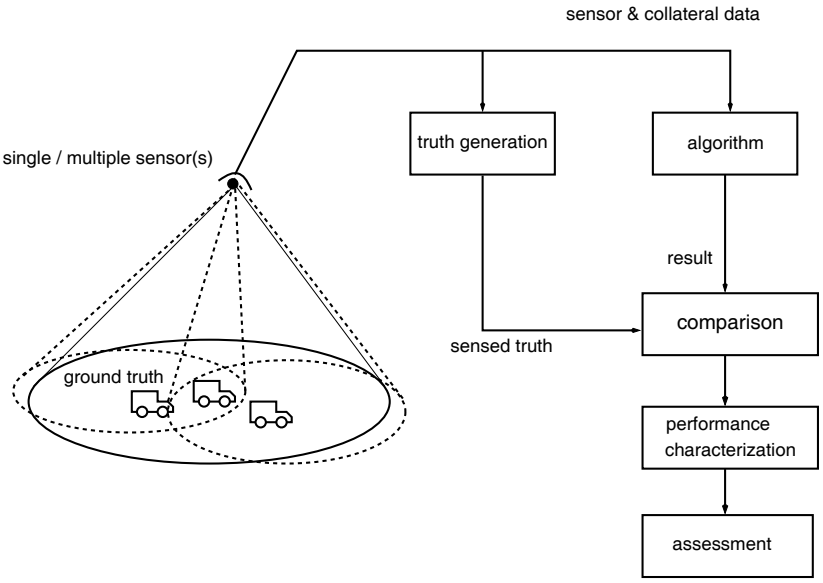

Figure 7.1: *Illustration of ground truth, sensed truth and algorithm assessment.*

Note that ground truth data may also describe objects that are invisible in the sensor images. One reason for this is occlusion, for example, cars occluded by trees. Another reason for objects missing in sensor images is illustrated in Fig. 7.1: the scene is captured by one or several sensors with different fields of view represented by three ellipses of different size. The sensor field of view depicted by a solid ellipse contains all objects present in the scene, while the two dashed ellipses contain only two objects each.

A problem that is not illustrated in Fig. 7.1 is due to perturbations that may cause algorithm failures. Knowledge about these perturbations is required for a detailed analysis of algorithm performance.

Because of the cited problems we assign a sensor-specific description to each image. This sensor-specific description is referred to as *sensed truth* in Fig. 7.1.

The sensed truth contains only those objects that are actually captured by the sensors, along with a list of potential sources of misinterpretations. The sensed truth represents the ideal result of a computer vision algorithm. In the philosophy of the assessment concept presented in this contribution, sensed truth has to be generated interactively by a human interpreter. In order to establish the list of potential sources of misinterpretations, a verification in the scene may be required after a check of the acquired images.

7.3.3 Image data acquisition

The algorithm performance in general is not independent of scene parameters and parameters of the image acquisition process. For this reason the images used for performance characterization must be representative for the application if the measured performance is required to be representative. The problem that arises involves the number of parameters of the imaging process. Relevant parameters are, for example, scene and object parameters, parameters of sensor and sensor carrier, time of day, time of year and weather conditions. Generally, the space spanned by the relevant parameters is too large to be covered entirely by the test images. In practice a compromise must be made to concentrate on the most important parameters regarding the intended application and to acquire a subset of all possible images.

The data acquisition may be expensive and the generation of truth is time consuming. As empirical tests are expensive joint tests are necessary. They allow the exploitation of the resources of several institutions, academia and industry, in order to define and perform the tests, including the preparation of truth, the necessary calibration of the system, the huge amount of repeated measurements, and the proper analysis [2]. These are some of the advantages of establishing an assessment center.

7.3.4 Provision of training data for development

The stated lack of formal models for the imaging process implies that the specification of a computer vision algorithm cannot be done formally, but must be completed by real images of application-relevant scenes. For this reason the relevant data set is split into a test data set and a training data set. The training data set must be representative of the entire data set. Using this data set, the software developer may adapt his or her algorithm in order to obtain an optimal result for the given application.

7.3.5 Algorithm performance characterization

The steps preceding performance characterization are shown in Fig. 7.1. The algorithm is run on the images (sensor data) whereas for each image the collateral data are made available to the algorithm as well.

The succeeding step is the comparison between algorithm results and sensed truth descriptions. For this comparison, a similarity metric has to be defined. Depending on the type of information that the algorithm has to extract from the image (algorithm class), different similarity measures may be appropriate:

- result point inside/outside truth polygon;

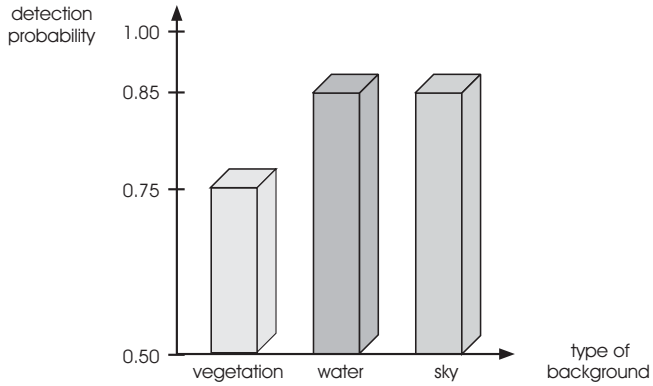

Figure 7.2: *Example of a performance profile.*

- correlation coefficient;
- sum of squared differences;
- contour difference; or
- structural matching distances, etc.

The computed differences and distances between algorithm results and sensed truth are then processed to obtain the actual algorithm performance criteria.

For a detailed insight we represent the algorithm performance over the space that is spanned by the image acquisition parameters and scene parameters such as background, object type, and movement, etc. This representation is called the *performance profile* of the algorithm. An example for the performance profile of a detection algorithm is the representation of its detection probability (performance criterion) over object background (scene parameter) (see Fig. 7.2).

The performance profile is an important piece of information for the user of the computer vision algorithm because it contains detailed information about the expected algorithm performance under different operational conditions.

7.3.6 The assessment function

This section describes how an application-specific figure of merit can be obtained from the performance profile consisting of the measured performance criteria $k_1, k_2, \ldots, k_n$. The computation is done by means of the assessment function described in the requirement profile. As the requirement profile also defines some required values for the performance criteria, the FOM could be regarded as a measure of how well

the algorithm performance satisfies the requirements of a given application.

Formally, the FOM is the result of applying the *assessment function* using the measured performance criteria as parameters

$$\text{FOM} = f(k_1, k_2, ..., k_n) \tag{7.1}$$

In a simple approach, an assessment function could weigh each criterion k_i according to its costs C_i defined in the requirement profile and the FOM would be defined by summing the weighted criteria

$$\text{FOM} = \sum_{i=1}^{n} C_i \, k_i \tag{7.2}$$

Note that the FOM may also be represented as a function of the algorithm parameters that have an impact on the algorithm performance. As the assessment function is part of the requirement profile and as the obtained FOM is crucial for the decision regarding algorithm utility, it has to be stressed again that the requirement profile needs an agreement among user, developer and assessing institution.

7.3.7 Overview of the assessment procedure

Algorithm developers and users together with the assessing institution specify the application task from which an assessment task has to be derived. Figure 7.3 outlines the information flow involved in the assessment of a computer vision algorithm.

First, the application is analyzed to obtain relevant *scene and acquisition parameters*, and to define the *performance criteria* together with their *weightings* that determine the *assessment function*. A comprehensive assessment requires a complete acquisition of the scene parameters and their variation. However, the specific combinatorial relationships usually do not demand the processing of all variations. Rather, the strategy must be to restrict the assessment to the most relevant cases. The set of scene and acquisition parameters specifies the procedure of *data acquisition*. The resulting *assessment data* consist of the images taken by the sensor(s), the corresponding collateral data, and the *sensed truth* that has to be generated application-specific.

The *training data* (a small but representative subset of the assessment data) are delivered to the developer for *algorithm training*. After training the algorithms they are run on the remaining assessment data, called the *test data*. It is worth mentioning that during the algorithm test the algorithms have access only to image and collateral data but not, of course, to the truth data (*truth data extraction*).

Afterwards, the *algorithm result* is compared to the ideal result (the *truth*) by performing a *distance measurement*. A subsequent *statistical analysis* yields the *performance profile* of the algorithm. During

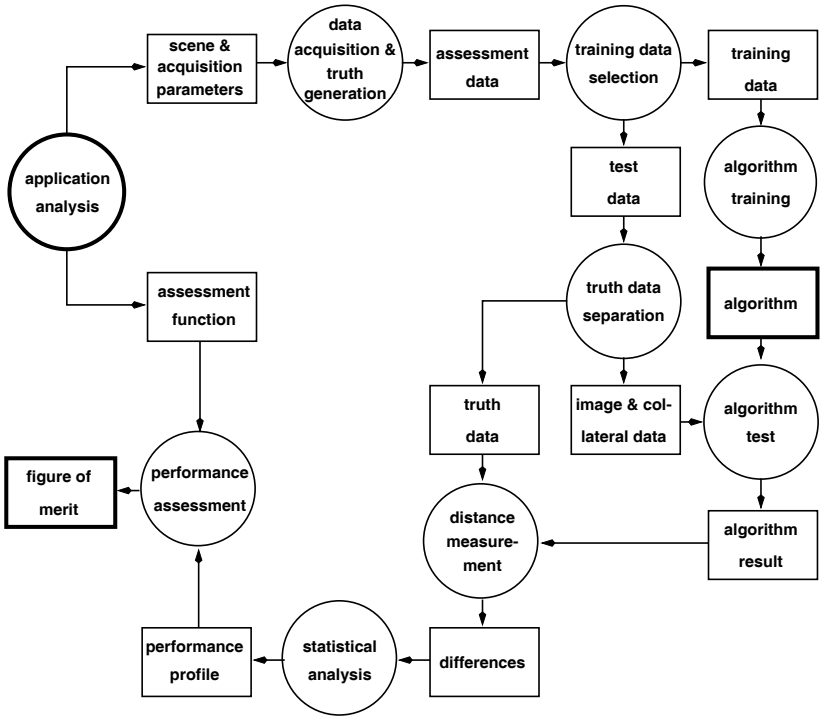

Figure 7.3: *Overview of steps involved in the assessment procedure. Boxes represent data, circles represent processes. The assessment of an algorithm begins with the application analysis and results in the figure of merit (FOM). These three major elements in the assessment procedure are highlighted.*

the *performance assessment* the application-specific FOM is computed from the algorithm performance profile by means of the assessment function. In the frame of an assessment strategy several competing algorithms may be evaluated and the best one is chosen for the application on the basis of the FOM.

7.4 The assessment system

Some, but not all steps of algorithm assessment can be executed automatically by means of a software tool collection [40, 41]. For example, the application of an algorithm to the test data would be a tideous task without the help of such a software. Likewise, managing large test data sets can essentially be simplified by a tool that provides management functionality in combination with a database. Hence, a toolbox was developed to support algorithm analysis and assessment, an overview of which will be given in this section.

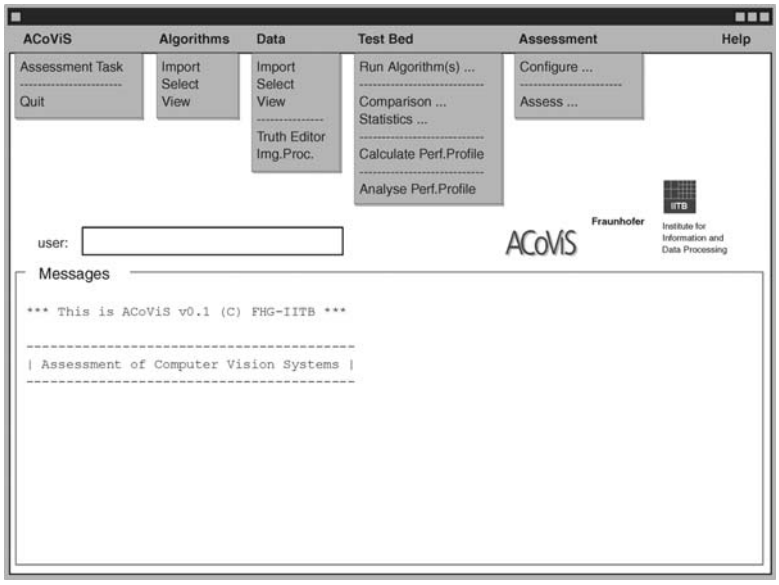

Figure 7.4: *The graphical user interface of the assessment system with menues pulled down.*

Revising the concept described in the foregoing sections one may identify five main topics that have to be addressed by the assessment system:

- control and configuration of the assessment task;
- management of test data;
- management of tested algorithms;
- performance measurement (*test bed*); and
- performance assessment.

It was a major goal of the design stage to reflect these steps on the top of the assessment system, which resulted in the graphical user interface shown in Fig. 7.4. The underlying functionality that is accessible through the menu entries will be outlined in the following.

Assessment Task. As stated in the foregoing, the main viewpoint for the assessment of computer vision algorithms is the intended use of an algorithm. The application essentially determines the parameters of the assessment, for example, which algorithms are available for the given application, which images are available and how the algorithm performance is to be assessed. Together with the requirement profile the application has to be formalized as an *assessment task* and com-

mitted to the system. This is accomplished by means of an input sheet where all of the relevant parameters are to be stated.

Data management. An essential part of a software system for algorithm testing is the management of data. There are two aspects to mention: first, the large number of images necessary to yield statistically significant performance measures; and second, the variety of the images concerning their content and formation (sensor, scene, mapped objects, etc.) required for differerent applications. Especially the latter is crucial for data management: Every image should be stored together with a description that allows a selection of images according to some properties relevant for a given application. Finally, sensed truth has to be provided for every image as well.

The following three functions of the assessment system are available for data management: *import* of newly acquired data, *selection* of data to accomplish a given assessment task; and statistical analysis (*view*) of test data in order to check for statistically significant occurrences of application-relevant image content.

Algorithm management. For the management of algorithms a function comparable to that of data management is provided. First, an *import* tool provides for the gathering of an algorithm together with its specification and parameters. Next, the *selection* of algorithm(s) to be tested is carried out according to a given application. A tool to *view* the specification of a stored algorithm provides detailed information about the algorithm properties.

Test bed. The steps that are supported by the test bed of the assessment system are drafted in Fig. 7.3: execution of the algorithm using the available test data; comparison between algorithm results and truth, using an appropriate distance measure; computation and analysis of algorithm performance profiles.

Assessment. The crucial step of the assessment concept described in this contribution is the *assessment* of the algorithm performance regarding the underlying application. Two tools are provided for editing an assessment function and to perform the assessment and to display the results.

7.5 Example: Assessment of a detection algorithm

This section shows how to assess a vehicle-detection algorithm in the context of two different applications. For both applications the algorithm task consists in detecting vehicles in infrared imagery.

The considered algorithm returns the image coordinates of the centroid of detected vehicles. Additionally, for each centroid the algorithm calculates a number indicating its own confidence into the produced result. This number is called the *score* of the algorithm result.

The truth for each vehicle is given by a surrounding polygon in image coordinates. The comparison between algorithm results and truth polygons consists in checking if the result centroids are inside the corresponding truth polygons. The following cases may occur:

- One and only one result centroid is in a truth polygon (*detection*);
- More than one result centroid is in a truth polygon (*multiple detection*);
- A truth polygon exists without a corresponding result centroid (*miss, nondetection*);
- A result centroid is outside of all truth polygons (*false alarm*); and
- No result centroid is generated by the algorithm given an image without any truth polygon (*background*).

For each case the corresponding *rate* is calculated by normalizing the number of occurrences. In this example, the detection rate, nondetection rate and the multiple detection rate were calculated by normalizing the corresponding numbers by the number of objects. The background rate is calculated by normalization to the number of images and the false alarm rate by normalization to the number of events, which is the sum of all detections, multiple detections, misses, false alarms and backgrounds.

To gain insight into the algorithm behavior with respect to the score threshold, one can calculate the mentioned rates while ignoring events whose scores are smaller than a given *score threshold*. By varying the score threshold, one obtains a plot similar to the one in Fig. 7.5, which shows the detection rate vs the score threshold.

One major problem is how to build from these simple performance measures a FOM indicating the utility of the algorithm with regard to the application. This was done by the construction of an assessment function R working as a criterion that has to be optimized by variation of the score threshold s

$$R(d,f,n,b,m;s) = C_d\,d(s) + C_f\,f(s) + C_n\,n(s) + C_b\,b(s) + C_m\,m(s) \tag{7.3}$$

where $C_i \in [-1,1]$, $i \in \{d,f,n,b,m\}$, and $d(s)$ is the detection rate, $f(s)$ is the false alarm rate, $n(s)$ is the nondetection rate, $b(s)$ is the background rate, and $m(s)$ is the multiple detection rate.

The constants C_i are application-dependent *cost* factors, which are used to address the relative importance of each event-type. A negative cost factor indicates a benefit, and a positive cost factor indicates a

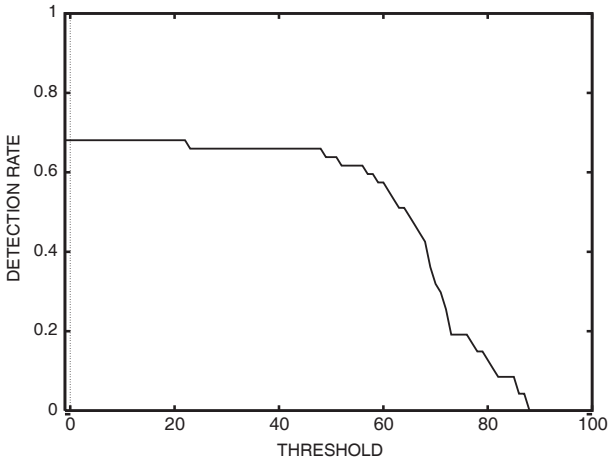

Figure 7.5: *Detection rate over the score threshold s.*

penalty. In respect thereof, $R(d, f, n, b, m; s)$ can be interpreted as the *risk* of applying the algorithm while rejecting results with score values smaller than s. In the underlying example, the FOM is defined as follows:

$$\text{FOM} = \min_{s} R(d, f, n, b, m; s) \tag{7.4}$$

It takes into account the weighted contribution of each performance criteria over the range of the score threshold s (see Fig. 7.6) and seeks the result with the lowest risk.

Using qualitative cost factors one can illustrate the assessment principle. The two applications considered here are:

- an *information gathering system (IGS)*, in which the algorithm results are further processed by human operators; and
- an *autonomous system (AS)*, in which decisions of the algorithm are directly taken without human interaction.

The cost factors are determined by applying the following considerations:

- detections/background: correct decisions that are rewarded in both applications (negative costs, cost factor -1.0);
- definitely, nondetections are counterproductive to the algorithm task. Therefore, they are penalized by a cost factor of 1.0 for both applications;
- multiple detections may lead to an incorrect evaluation of the observed situation in the IGS. Therefore the multiple detection rate is weighted with a cost factor of 0.7; for an AS the crucial information, for example, for obstacle avoidance, is the position of detected

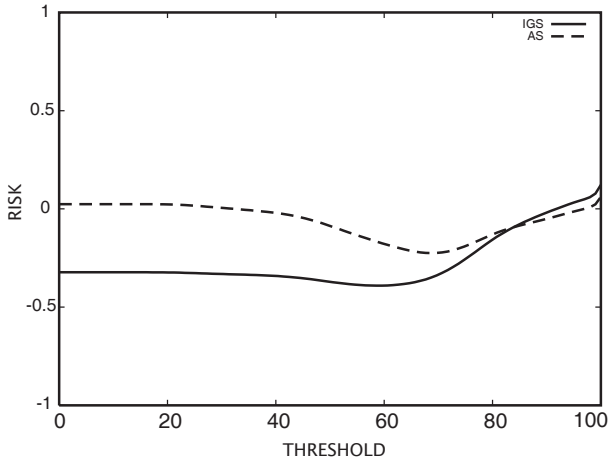

Figure 7.6: *Assessment function of the information gathering system (IGS, solid line) and the autonomous system (AS, dotted line). The minimum indicates the FOM and the task-specific operation point.*

objects. Multiple detection hypotheses are less critical for this application because they do not lead to incorrect decisions. The cost factor is therefore chosen to be 0.1;

- false alarms may lead to incorrect decisions in the AS and are therefore penalized by a factor of 1.0. In the interactive application a false alarm can be rejected by a human operator and thus is less problematic. The cost factor is chosen to be 0.5.

As already mentioned in the foregoing, the threshold applied to the scores of the detection hypotheses is used to reject weak algorithm results. The impact thereof on the risk depends on whether the rejected result is a detection, a multiple detection, a miss, a false alarm, or a background. For instance, the rejection of multiple detections and false alarms reduces the risk, whereas rejected detections and backgrounds increase the overall risk of the detection algorithm.

Figure 7.6 shows a plot of the risk $R(d, f, n, b, m, s)$ for both applications over the variation of the corresponding score threshold. The IGS has a minimum risk at a threshold of 60 indicating the optimal operation point. By contrast, the optimal operation point for the autonomous system is achieved at a score threshold of 70.

Further, the minimum risk of using the algorithm in the interactive application less than the minimum risk of using it in the autonomous application, which means that the algorithm has a higher utility for the IGS than for the autonomous system.

7.6 Conclusion

The assessment of computer vision algorithms is more than just a question of statistical analysis of algorithm results. Rather, the algorithm field of application has to be taken into account as well. Assessment as understood in this contribution quantifies the utility of image analysis algorithms with respect to given applications. A clear distinction between performance analysis and performance assessment has to be made. The underlying algorithm performance is determined experimentally. The final statement, how much an algorithm fulfills its requirements, demands basic knowledge about its application-dependent context. The connection between algorithm performance and this context is modeled by an application-dependent assessment function.

Acknowledgments

This work was funded by the German Federal Office for Defense Technology and Procurement under Project E/F41B/V0037/Q5242. The authors wish to thank B. Dürr, Dr. U. Jäger, and G. Saur from the Fraunhofer-Institut für Informations- und Datenverarbeitung, Karlsruhe, Germany for many helpful discussions and suggestions. The detection algorithm was developed by M. Müller, L. Berger, M. Esswein, N. Heinze, Dr. W. Schwerdtmann, and C. Sung (all from the Fraunhofer-Institut für Informations- und Datenverarbeitung). G. Hofele from the Forschungsinstitut für Informationsverarbeitung und Mustererkennung, Ettlingen, Germany provided the data required for the assessment example.

7.7 References

[1] Förstner, W., (1994). Diagnostics and performance evaluation in computer vision. In *Proceedings of the Workshop on Robust Computer Vision*. Seattle, USA.

[2] Förstner, W., (1996). 10 Pros and cons against performance characterisation of vision algorithms. In *Proceedings of the ECCV Workshop on Performance Characteristics of Vision Algorithms*. Cambridge, UK. http://www.vision.auc.dk/ hic/perf-proc.html.

[3] Haralick, R., (1994). Overview: Computer vision performance characterization. In *Proceedings of the ARPA Image Understanding Workshop*, pp. 663–665. Monterey.

[4] Haralick, R. M., (1994). Performance characterisation protocol in computer vision. In *Proceedings of the ARPA Image Understanding Workshop*, pp. 667–673. Monterey.

[5] Jain, R. C. and Binford, T. O., (1991). Ignorance, myopia, and maiveté in computer vision. *Computer Vision, Graphics, and Image Processing: Image Understanding*, **53**(1):112–117.

[6] Zamperoni, P., (1996). Plus ça va, moins ça va. *Pattern Recognition Letters*, **17**:671–677.

[7] Förstner, W., (1987). Reliability analysis of parameter estimation in linear models with applications to mensuration problems in computer vision. *Computer Vision Graphics and Image Processing*, **40**(3):273–310.

[8] Haralick, R. M., (1994). Propagating covariances in computer vision. In *Proceedings of the International Conference on Pattern Recognition*, pp. 493–498. Jerusalem, Israel.

[9] Bar-Shalom, Y. and Li, X., (1993). *Estimation and Tracking*. Boston: Artech House.

[10] Demigny, D. and Kamlé, T., (1997). A discrete expression of Canny's criteria for step edge detector evaluation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, **19**.

[11] Fuchs, C., Lang, F., and Förstner, W., (1997). On the noise and scale behaviour of relational descriptions. In *DAGM-Workshop on Performance Characteristics and Quality of Computer Vision Algorithms*. Braunschweig, Germany.

[12] Salotti, M. and Garbay, C., (1996). Evaluation of edge detectors: critics and proposal. In *Proceedings of the ECCV Workshop on Performance Characteristics of Vision Algorithms*. Cambridge, UK.
http://www.vision.auc.dk/~hic/perf-proc.html.

[13] Spreeuwers, L. and van der Heijden, F., (1992). Evaluation of edge detectors using average risk. In *Int. Conf. on Pattern Recognition (ICPR92)*, Vol. 3, pp. 771–774. The Hague.

[14] Conners, R. W. and Harlow, C. A., (1980). A theoretical comparison of texture algorithms. *IEEE Trans. Pattern Recognition and Machine Intelligence*, **2**(3):204–222.

[15] de Graaf, C., Koster, A., Vincken, K., and Viergever, M., (1992). Task-directed evaluation of image segmentation methods. In *Proc. Int. Conf. Pattern Recognition*, Vol. 3, pp. 219–222. The Hague.

[16] Wang, Z., Guerriero, A., and De Sario, M., (1996). Comparison of several approaches for the segmentation of texture images. *Pattern Recognition Letters*, **17**:509–521.

[17] Lindenbaum, M., (1997). An integrated model for evaluating the amount of data required for reliable recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence Systems*, **19**(11):1251–1264.

[18] Sarachick, K., (1994). The effect of Gaussian error in object recognition. In *Image Understanding Workshop*, Vol. 2. Hyatt Regency Hotel, Monterey, California.

[19] Conners, R. and Ng, C., (1989). Developing a quantitative model of human preattentive vision. *IEEE Trans. Systems, Man, and Cybernetics*, **19**(6):204–222.

[20] Shirvaikar, M. and Trivedi, M., (1992). Developing texture-based image clutter measures for object detection. *Optical Engineering*, **31**(12):2628–2639.

[21] Waldman, G., Wootton, J., Hobson, G., and Luetkemeyer, K., (1988). A normalized clutter measure for images. *Computer Vision, Graphics, and Image Processing*, **42**:137–156.

[22] Schmieder, D. and Weathersby, M., (1983). Detection performance in clutter with variable resolution. *IEEE Trans. Aerospace and Electronics Systems*, **19**(4):622–630.

[23] Ratches, J., Walters, C., Buser, R., and Guenther, B., (1997). Aided and automatic target recognition based upon sensory inputs from image forming systems. *IEEE Trans. Pattern Analysis and Machine Intelligence*, **19**(9): 1004–1019.

[24] Weszka, J., Dyer, C., and Rosenfeld, A., (1976). A comparative study of texture measures for terrain classification. *IEEE Trans. Systems, Man and Cybernetics*, **6**:269–285.

[25] Kanungo, T., Jaisimha, M., Palmer, J., and Haralick, R. M., (1995). A methodology for quantitative performance evaluation of detection algorithms. *IEEE Trans. Image Processing*, **4**(12):1667–74.

[26] Haralick, R., (1989). Performance assessment of near-perfect machines. *Machine Vision and Applications*, **2**:1–16.

[27] Bradley, A. P., (1996). ROC curves and the χ^2 test. *Pattern Recognition Letters*, **17**:287–294.

[28] Haralick, R., (1996). Detection performance methodology. In *ARPA Image Understanding Workshop*, pp. 981–983. Palm Springs, USA.

[29] Liu, X., Kanungo, T., and Haralick, R., (1996). Statistical validation of computer vision software. In *ARPA Image Understanding Workshop*, pp. 1533–1540. Palm Springs.

[30] Takeshita, T. and Toriwaki, J., (1995). Experimental study of performance of pattern classifiers and the size of design samples. *Pattern Recognition Letters*, **16**:307–312.

[31] Nyssen, E., (1996). Evaluation of pattern classifiers—testing the significance of classification using an exact probability technique. *Pattern Recognition Letters*, **17**:1125–1129.

[32] Cho, K., Meer, P., and Cabrera, J., (1997). Performance assessment through bootstrap. *IEEE Trans. Pattern Analysis and Machine Intelligence Systems*, **19**(11).

[33] Jain, A. K., Dubes, R. C., and Chen, C.-C., (1987). Bootstrap techniques for error estimation. *IEEE Trans. Pattern Recognition and Machine Intelligence*, **9**(5):628–633.

[34] Chernick, M., Murthy, V., and Nealy, C., (1985). Application of bootstrap and other resampling techniques: Evaluation of classifier performance. *Pattern Recognition Letters*, **3**(3):167–178.

[35] Courtney, P., Thacker, N., and Clark, A., (1996). Algorithmic modelling for performance evaluation. In *Proceedings of the ECCV Workshop on Performance Characteristics of Vision Algorithms*. Cambridge, UK. http://www.vision.auc.dk/ hic/perf-proc.html.

[36] Rivlin, E., Aloimonos, Y., and Rosenfeld, A., (1991). *Purposive Recognition: A Framework*. Technical Report CAR-TR-597, Center for Automation Research, University of Maryland.

[37] Stilla, U., (1997). Automatische Extraktion von Gebäuden aus Luftbildern mit Produktionsnetzen. In *DGPF-Jahrestagung, Arbeitskreis Bildanalyse*.

[38] Stilla, U., Michaelsen, E., and Lütjen, K., (1996). Automatic extraction of buildings from aerial images. In *Mapping Buildings, Roads and Other Man-Made Structures from Images*, F. Leberl, R. Kalliany, and M. Gruber, eds. Wien: Oldenburg.

[39] Huertas, A., Bejanin, M., and Nevatia, R., (1995). Model registration and validation. In *Automatic Extraction of Man-Made Objects from Aerial and Space Images*, A. Gruen, O. Kuebler, and P. Agouris, eds., pp. 33–42. Basel.

[40] Clark, A. F., (1996). HATE: A Harness for Algorithm Testing and Evaluation. Available from http://peipa.essex.ac.uk/hate/.

[41] De Boer, C. and Smeulders, A., (1996). BESSI: An experimentation system for vision module evaluation. In *International Conference Pattern Recognition ICPR96*, p. C70.2. Vienna, Austria.