Graz University of Technology

Institute for Computer Graphics and Vision

Master's Thesis

Computer-Vision based Pharmaceutical Pill Recognition on Mobile Phones

Andreas Hartl

Graz, Austria, February 2010

Supervisor:
Univ.-Prof. DI Dr.techn. Dieter Schmalstieg

Advisor:
DI Dr.techn. Clemens Arth

Abstract

In this work we present a mobile computer vision system which simplifies the task of identifying pharmaceutical pills. A single input image of pills on a special marker-based target is processed by an efficient method for object segmentation on structured background. Estimators for the object properties size, shape and color deliver parameters that can be used for querying an on-line database about an unknown pill. A prototype application is constructed using the Studierstube ES framework, which allows to carry out the entire procedure of pill recognition on off-the-shelf mobile hardware. For the purpose of pill retrieval, an additional piece of software is introduced which runs on an ordinary web server. It may deliver preprocessed pill information from an arbitrary database to the mobile device and serves as an interface for arbitrary sources of information. The performance of the estimators as well as their runtime is subsequently evaluated with conditions that resemble typical environments of use. The retrieval performance on the exemplarily used Identa database confirms that the system can facilitate the task of mobile pill recognition in a realistic scenario.

Acknowledgments

I want to thank Dieter Schmalstieg for giving me the opportunity to work on this project. I also want to thank Clemens Arth for giving a great deal of beneficial input and providing me with mobile hardware for testing. His steady support in technical as well as administrative issues made it possible to carry out this project. Additional thanks go to Daniel Wagner, Lukas Gruber and Tobias Langlotz from the Institute of Computer Graphics and Vision for their valuable hints on application development in Studierstube ES.
Above all I would like to thank my lovely family for their ongoing support during my studies. Special thanks go to my brother Stefan, who encouraged my effort in this project and supported me in times of temporary confusion. Last but not least, I want to thank all my friends who had the doubtful honor to bear my gloom, when things looked rather dark.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, February 2010

Andreas Hartl

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

The pharmaceutical industry is an enormous branch of economy, primarily in the western society. Costs for healthcare are permanently increasing and drugs have a considerable impact on them. It is desirable to provide a suitable therapy at tolerable cost, which is a challenge both for physicians and economists. Correct and timely identification of drugs constitute a key factor in an efficient healthcare infrastructure. The benefit of a systematic identification is the ability to apply a more systematic therapy or even to avoid unnecessary medical effort. The following scenarios emphasize the need for an authentic identification [28]:

- Intoxication of the patient.
- Cases where the patient is in a state of unconsciousness, under the influence, or confused.

Often, the process of identification must be carried out in a mobile scenario, where, in the absence of other tools, only a visual inspection is possible. A contribution to tackle this problem may be found in the application of techniques from the field of computer vision on ubiquitous mobile devices. The growing processing power of these devices, advancements in connectivity and the availability of an integrated camera make it possible to create a mobile system that supports the process of visual identification of pills, which is the topic of this work.

1.2 Mobile Phones and Smartphones

Nowadays mobile phones are seamlessly integrated into the daily routine. They have continuously inherited functionality from other mobile devices. Modern mobile phones may not only serve as communication facility but also as personal information manager, digital still camera, audio player, navigation device, handheld television or gaming platform.

Very advanced mobile phones that offer more and possibly novel features when compared to the mainstream models, are generally called smartphones. They include more powerful processing units, improved displays, extended possibilities of user input, as well as communication facilities. Additional hardware such as a dedicated GPU, accelerometer or GPS receiver may also be integrated. Advancements in clock frequency as well as architecture allow the creation of fast responding devices that can run demanding applications at reasonable speed. The fitted displays increase in size, resolution as well as color depth and may possibly serve as input device (touchscreen). Built-in cameras may feature automatic focus, video light and deliver still images and video of good quality. In general, a large number of communication facilities is supported. They can be divided in such requiring a service provider and those which can be used in a local environment (including Wireless Local Area Network).
Although the term *smartphone* is frequently used nowadays, a commonly agreed definition is still lacking.

Manufacturers of advanced mobile hardware often use a third party operating system and encourage the creation of new applications by providing software development kits (SDKs) to the public. Currently the smartphone market is dominated by four major software platforms [6]. These are Symbian OS[1], Blackberry OS[2], iPhone OS[3] and Windows Mobile[4]. Popular devices are for example the Nokia N96[5], the BlackBerry Storm[6], the Apple iPhone (3GS), the Samsung Omnia SGH-i900[7] and the Asus M530w[8] (see Figure 1). Although Symbian based platforms have most market share, the fast growing Google *Android*[9] platform may change matters in the medium term.
In the second quarter of 2009, more than 38 million smartphone devices have been shipped worldwide. This corresponds to an increase of 13.4% when compared with the same period in 2008 [6]. This increasing propagation of devices makes application development more attractive and a broader audience may benefit from new developments. For the remainder of this work, the term *smart-*

Figure 1: Popular smartphones (relative scale is not representative): A...Nokia N96, B...BlackBerry Storm, C...Apple iPhone 3GS, D...Samsung Omnia SGH-i900, E...Asus M530w

[1] http://developer.symbian.org
[2] http://de.blackberry.com/developers
[3] http://developer.apple.com/iphone
[4] http://developer.windowsmobile.com
[5] http://europe.nokia.com/find-products/devices
[6] http://na.blackberry.com/eng/devices
[7] http://omnia.samsungmobile.com
[8] http://www.asus.com
[9] http://developer.android.com

phone refers to an advanced mobile phone, where the manufacturer intends and supports the creation of new applications by providing suitable software interfaces and development tools.

1.3 Problem Definition

The task is to build a mobile computer vision system that makes the visual identification of pharmaceutical pills a less tedious task. Extracted features from a *single image* should be used to back up a query towards an existing database. The system should be able to operate on a Windows Mobile based smartphone with fixed-point arithmetics, at reasonable speed. The system should be developed using C++ within the Studierstube ES [52] framework for handheld augmented reality.

1.3.1 Requirements

- *Robustness:* The recognition module should be able to operate under a wide range of conditions e.g.: rotation, scale, type of ambient light.
- *Usability:* A system that is easy to use can be operated by a broader audience. The aim is to take away the burden of complicated input from the user.
- *Efficiency:* The problem should be computable on fixed-point platforms in a reasonable amount of time. Besides, the amount of transmitted data should be kept as small as possible.

1.4 Classification of Drugs

Drugs can be divided into solid, semisolid, liquid and aeriform types, depending on their state and stability of shape (see the work of Estler et al. [18]). They can be further classified into subtypes, which are adapted to a specific application. A well known example can be found in enteric-coated oral drugs that are able to protect acid-labile agents. The patient gets the drug possibly in a form that may change before intake (e.g.: effervescent tablet). When we refer to the form the patient gets within this work, we use the word *presentation*. A listing of drug classes with their subtypes is shown in Table 1.
Oral drugs have most market share nowadays, since they can be taken conveniently, are easy to manufacture and long-living. Out of the solid group, the recognition of tablets, capsules and pills (dragées) is a critical task that must often be performed as fast as possible. Whenever we refer to the word *drug* or *pharmaceutical pill* within this work, we mean the aforementioned subtypes.

Solid	Semisolid	Liquid	Aeriform
powder	unguents	injections	particulate matter
granulate	cream	infusion	
tablet	gel	drops	
capsule	paste	lotion	
pill	plaster	extract	
implant		juice	
suppository		syrup	
globules		tea	

Table 1: Classification of drugs (translated from [18])

1.5 Identification of Pharmaceutical Pills

1.5.1 Domain and Means of Identification

Some domain research reveals three main methods for pill identification:

- Look-up in a book.
- Application of a special identification scheme.
- Querying a database.

The book method can be applied with literature like *The Physician's Desk Reference* [17]. It contains information about a large number of pills. Besides name and manufacturer, additional information is available (e.g. contents, dose, effectiveness). Example pictures of pills are provided, which can be helpful. This means that the book must be at hand when an identification is necessary. Besides, it might take some time to find the right pill in such a protruding piece of literature. Medical knowledge, at least up to a certain degree, may also help in the process.
Another possibility is to apply an identification scheme. In Austria, the first scheme that was not restricted to forensic use, was created in the year 1999 [28]. It is called *Schnellerkennung* (translated as *rapid identification*) and allows recognition of tablets, coated tablets and capsules, mostly by inspection of their external properties. It aims to identify a preparation that is encountered without its packaging with low effort concerning time and cost. This scheme of identification is stated as "applicable by people that are not specifically trained". The kit fits within a folder and includes the scheme, a nonius for size measurements and color charts. Features of interest are as follows:

- *General Properties:* type of pharmaceutical pill
- *Outer Properties:* shape, color, size, scores, imprint or embossment
- *Inner Properties:* mass, consistency, outcome of certain chemical reactions

In some cases, simple chemical experiments may be necessary to find an exact match, however. As of the introduction of the scheme, it was claimed to be able to identify all pills available on the market.
A method that employs information technology is the directed query of a special database with pill information. In general, the query parameters are a subset of the features used in the rapid identification scheme. This translates to a loss in

accuracy, since an exact match will not be possible in every case then. It would of course be possible to incorporate all information of the rapid identification scheme into a an information system, but such a solution does not exist or is not accessible.
Research in the area reveals three major candidates for database queries. These are *Drugs.com*[10], *Epocrates Pill Identifier*[11] and *Identa*[12](see Table 2). It is important to note that out of these databases, the first and the last one allow free access. There is an additional database called *Drug Identity Information*[13],

Name	Pres.	Shape	Color	Size	Score	Impr.	Coat.	Clar.	Mass
Drugs.com		x	x			x			
Epocrates		x	x		x	x	x	x	
Identa	x	x	x	x					x

Table 2: Pill identification databases and properties

that relies on a series of questions for identification. The exact features, however, cannot be determined without pay. If the rapid identification scheme is considered as the baseline for further considerations, it is evident that it is not possible to identify all pharmaceutical pills on the market with features that can be obtained by visual inspection (including size measurements). The expected accuracy will even be below the database solution, since certain features are not supported.
In Table 3 the methods outlined above are presented with a rating for their accuracy, speed, availability and necessary training of medical staff.

Method	Accuracy	Speed	Availability	Training
book	good	low	low	medium
scheme	very good	medium	low	medium
database	good	good	low	medium

Table 3: Comparison of identification methods

1.5.2 Computer Vision Aided Identification

An investigation of samples and query parameters reveals several properties for pill objects that could be determined by techniques of computer vision: These are the object shape, size, color, scores and imprints. For the majority of pharmaceutical pills, these features show the following characteristics:

- Shape: convex and symmetric
- Size: length, width, height
- Color: one or two colors (border follows major or minor axis)
- Scores: zero or more with variations in location
- Imprints: none, one or both sides

[10] `http://www.drugs.com`
[11] `https://www.epocrates.com`
[12] `http://www.gelbe-liste.de/pharmindex/identa`
[13] `http://www.drugid.info`

Besides, special forms exist that may differ in characteristics. A query towards Identa for example, returns about 50 results that have non-convex shape. Thus, for the remainder of this work, special forms in the sense that they differ from the assumed features and their characteristics, are not considered.

1.6 Thesis Outline

The remainder of this work is structured in the following way: In Section 2 a range of related work is presented that serves as a starting point for the design of the system. First an overview of computer vision on mobile devices is given. Then segmentation is studied and possibilities for the determination of object properties are investigated. Section 3 is devoted to the selection of a suitable segmentation technique, as it is a fundamental problem in the application to be developed. Section 4 deals with the extraction of object properties from the obtained regions. In Section 5 an overview of an exemplary StbES implementation for mobile devices is given that is able to connect to the Identa pill database. The created system is then evaluated in Section 6 using the available samples. We conclude this work in Section 7 with a summary of results and future prospects.

2 Related Work

2.1 Computer Vision on Mobile Phones

Important aims in computer vision are to *see* (segmentation, classification, identification) and *understand* (modeling, tracking) a given scene [34]. The involved tasks may require huge computational resources and very advanced algorithms. A possible target platform for computer vision applications are modern mobile phones and smartphones, since they include a built-in camera and many manufacturers support the creation of new applications. On the one hand mobile phones or smartphones offer rather limited resources for computation, because of cost and available energy. On the other hand they have the advantage of being available off the shelf, and are enjoying a very large and growing user base.

2.1.1 Challenges

From the perspective of the user it is desirable to build reliable and responsive software, which may be used intuitively [34]. Thus, a key factor for the success of a mobile computer vision application is the associated user experience. Mobile development is hampered by limitations of the target devices, however. Until recently, fixed focus cameras were common as image acquisition devices. With them it is not possible to take sharp images from objects at a low distance. Because of lacking standardization, developers are given no or only limited possibilities to influence the image acquisition process.
The processing units of mobile phones have lower clock rates, smaller cache sizes and are optimized for energy efficiency, resulting in lower overall performance. Except for some high end devices, floating point units are not available. Compiling unmodified code for a fixed-point platform forces the compiler to create emulation code which causes a huge degradation in performance. So, floating point calculations need to be avoided whenever possible [52]. Alternatively, fixed-point types can be used, which have a much smaller performance impact. Memory is limited both in size and bandwidth. Typically slower types are used, because they tend to consume less energy. The latter is also limited, as larger battery packs would mean higher cost and possibly reduce mobility.
An increasing number of manufacturers offer software development kits. They contain libraries, APIs [14] and device emulators that are intended to facilitate application development. In general, it is difficult to debug mobile applications requiring a camera when using these emulators on the development platform.

2.1.2 Studierstube ES

Studierstube ES[15] (StbES) is a component-based software framework for handheld augmented reality that supports cross-platform development using C++ [52]. It was developed in 2006 at the Institute of Computer Vision and Graphics (part of the Technical University) in Graz, Austria. StbES provides a high level API with convenient capabilities of platform abstraction and supports Windows CE/Mobile/2K/XP, Linux, Symbian and the iPhone (see Figure 2

[14] Application Programming Interface
[15] Embedded System

for an overview of the architecture[16]). Out of these, Windows Mobile is cur-

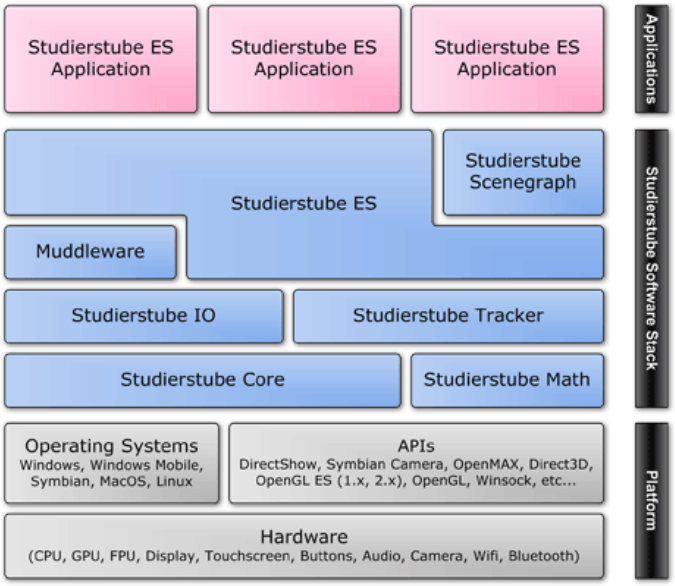

Figure 2: Architecture of the StbES framework

rently the main target platform. With StbES it is possible to create and debug camera-based applications on an ordinary personal computer running Windows or Linux.

The StbES framework is highly optimized for mobile hardware, features a small memory footprint and provides efficient fixed-point math classes (as part of Studierstube Math). All major pixel formats are supported, so that costly conversions can be reduced to a minimum.
Generation and output of 3D graphics is facilitated by a lightweight XML-based scene graph implementation (Studierstube Scenegraph) with dedicated rendering paths for hardware and software accelerated devices. Supported standards are OpenGL ES 1.x, OpenGL ES 2.x, OpenGL, and Direct3D Mobile. In addition, there is support for the creation of 2D graphical user interfaces (GUI).
An efficient and robust tracking framework (Studierstube tracker) is available in StbES and can serve as a basis for state-of-the-art computer vision and augmented reality applications. A variety of marker types is supported and access is possible trough a fully class-based API.
Networking support is targeted especially towards large numbers of users as may be the case in shared mobile augmented reality applications. Depending on the nature of the application, a connection to a server may be desirable (e.g. for holding global state). The Muddleware networking middleware was designed to satisfy these needs and can run standalone or with backup from a server. Besides, standard communication methods such as a HTTP [17] request are also available.

[16] `http://www.studierstube.org/handheld_ar`
[17] Hypertext Transfer Protocol

Applications for StbES are compiled as dynamic link libraries (DLL), which serve as a means of abstraction between application and framework. Necessary parameters must be set using configuration files in XML format. It is possible to configure StbES to start up, perform tracking and show graphical output before application code has been written. Configuration of camera and tracking parameters as well as scene graph creation may also be handled using these files. For certain applications it may be necessary to manually instantiate components like the tracker from within the application.

StbES may address several challenges in mobile development and serve as an efficient basis for the development of mobile computer vision applications.

2.1.3 Applications

Camera phones are an increasingly attractive platform for the development of mobile computer vision applications. At the beginning, they served mainly as image acquisition devices and the actual processing was carried out on a dedicated server. Since the topic of this work is a mobile application in which computer vision is performed on the device itself, this distributed type of application is not considered. In the following a series of mobile computer vision applications are presented.

Image alignment and stitching algorithms are widely used in computer vision (see e.g. the work of Szeliski [61] for a survey of methods). Besides frame alignment for image stabilization, the creation of panoramic images from a series images taken from a scene is a very popular application. Mobile approaches, which process images taken in an off-line step by the built-in camera are common. Recently a real-time approach for indoor and outdoor panoramic mapping and tracking using natural features has been implemented [18]. This application allows for on-line creation and simultaneous tracking of panoramas on modern mobile phones.

The identification of goods from a bar code is very common nowadays. Adelman et al. [1] created a recognition and resolving toolkit for EAN-13 bar codes in JAVA. It is available for free and may work on mobile phones equipped a camera featuring autofocus. All computer vision related processing is carried out on the mobile phone, whereas a server side component is used for the resolution of the code.
Computer vision based recognition of 2D visual codes is also available for mobile devices (see e.g. the work of Rohs [49]). A very popular application of 2D bar code recognition is *mobile tagging*[19], where usually an URL[20] is extracted from the code (see Figure 3). It may point to information such as weather forecasts or stock quotes on the Internet which may quickly get obsolete. This scenario may be a lot more attractive to the user, because it is no longer necessary to enter a complicated URL. It is also possible to extract information about tilt angle or the relative direction of movement (when several frames are considered). Rohs

[18] `http://www.studierstube.org/handheld_ar`
[19] `http://en.wikipedia.org/wiki/Mobile_tagging`
[20] Uniform Resource Locator

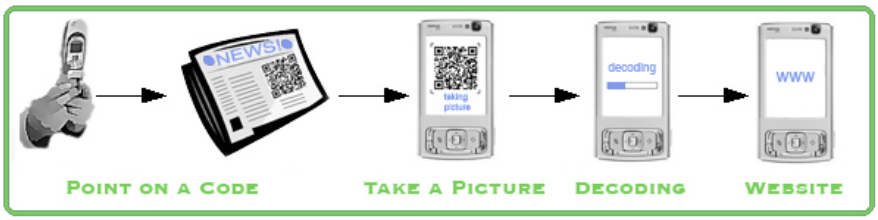

Figure 3: Illustration of *mobile tagging* (taken from Wikipedia)

[49] uses such information for interaction with large scale displays.

Urban intersections can be very dangerous for blind or visually impaired people. When crossing, the main problem is that of alignment with the crosswalk. Ivanchenko et al. developed a mobile computer vision system to provide real-time information about the location and orientation of crosswalks. It is called *Crosswatch* [33] and makes use of off-the-shelf mobile hardware. Whenever a

Figure 4: Crosswatch: providing real-time information about the location and orientation of crosswalks (taken from [33]).

crosswalk is detected and the mobile phone is aligned with it, an audio tone is given (see Figure 4). With this solution, no additional infrastructure such as audible pedestrian signals is necessary.

Computer vision on mobile phones may enhance the experience of people visiting a museum by replacing audio guides and providing additional information about exhibits. Föckler et al. developed a museum guidance system called *PhoneGuide*, that may work on camera-equipped mobile phones and performs on-device recognition of objects [23].
A different experience becomes possible when using smartphones as devices for augmented reality and packing the visit into an interactive game [52]. An example is called *Expedition Schatzsuche* (translated as *expedition treasure hunt*), which was developed for the Carinthia State Museum in Klagenfurt, Carinthia, Austria. Part of a visit may involve this interactive puzzle game, which was built on top of the Studierstube ES framework. Relevant exhibits are equipped with a marker and various tasks need to be solved by interacting with them. So, especially young visitors may explore part of the museum in a hands-on approach.

In the previous application visitors had the possibility to work together when solving the puzzles. Mulloni et al. have created a mobile augmented reality

game called *Cows vs. Aliens* in which mobility and social interaction among players are key elements [41]. The game objective is to look for some hidden items (cows) and collect them into a safe location (stable). Cows are placed on game locations corresponding to pastures and may be commanded to move between directly connected locations. Players may also command UFOs to annihilate cows so that the opposing team cannot save them anymore. The game

Figure 5: Augmented reality interface with 2D overlay from *Cows vs. Aliens* (taken from [41]).

is played in two teams, where each member is equipped with a smartphone and all game locations are visualized by augmenting markers (see Figure 5). Additional game information is visualized using 2D overlays. The result is a gaming experience in which device-mediated and real-world interaction paradigms are combined in a homogeneous manner.

The aforementioned applications have shown that there is tremendous potential for the application of computer vision on ubiquitous devices such as mobile phones. Despite advancements in computing power, limited resources for computations and storage of data pose a major challenge. So it is critical to take these constraints into account when designing mobile computer vision applications.

2.2 General Considerations

A study of related work that is also subject-specific reveals a US patent that describes a *pill recognition and verification system* [69]. The patent deals with larger scale prescriptions, as they occur in large hospitals or drug firms. It describes an automatic method for the task of verification that the content of a container filled by a dispensing system, corresponds to a given prescription.
The system uses a frame grabber as input and employs color, geometry and surface features to identify all pills that a given machine may process. The computed features are compared with reference data of processable pills on a particular machine. For the recognition task, a considerable amount of prior knowledge is available. The approach has the following properties, from the perspective of computer vision:

- Image acquisition: stationary system with fixed conditions; capture of

several frames

- Segmentation: probably edge based
- Geometry features: non-generic shape detector from binary edge maps; estimation of size
- Color features: cluster properties in LUV space (parallelepiped regions)
- Surface features: finish, texture, score, markings
- Prior knowledge: properties and amount of target pills are known

An incorporation of prior knowledge simplifies the problem and should ensure better results. For example, knowledge of the desired color suggests a suitable background for segmentation. Knowledge of the amount of colors allows to influence the classification step. As the desired amount of pills is known, segmentation is further simplified.

In the mobile pill recognition system that is to be designed and implemented within this work, the situation is different. The acquisition conditions are not fixed but are dependent on the current environment. So, pose as well as the lighting conditions may vary. These circumstances impede segmentation as well as the determination of object size. Aside from that, the problem must be solved with the computational constraints of a mobile device in mind.
For this reason, only those techniques are included in the following survey of possible solutions for the subtasks, that fulfill at least one of the following requirements:

- Low computational complexity may be expected.
- The amount of input data can be expected to be very small.
- A successful implementation exists in a related context.

In this context it seems reasonable to tackle the problem with a separated segmentation, feature extraction and classification step (depending on the feature), that may be optimized independently. Promising solutions may be found in the field of robot soccer, where segmentation and color classification with limited resources are topics of intensive research. In the following subsections each subproblem of the task is considered by a listing of possible techniques along with an explanation of the underlying concepts.

2.3 Survey of Segmentation Techniques

The main goal of segmentation is to divide an image into parts that have a strong correlation with objects or areas of the real world contained in the image [57]. It is desirable to achieve a complete segmentation and obtain a set of disjoint regions that uniquely correspond to objects in the input image. Segmentation of nontrivial images is one of the most difficult tasks in image processing. Its accuracy determines the eventual success or failure of computerized analysis procedures [25].

There are different approaches to segmentation [4]. In the *bottom-up* approach the image is segmented into regions first and then the regions that correspond to objects are identified. The main difficulty of this approach is that an object may be segmented into multiple regions, some of which may merge the object with its background. Another approach is called *top-down* and uses prior knowledge about an object to guide the process. The main difficulty in this approach lies in the large variability in the shape and appearance of objects within a given class. Thus, recognition may help segmentation and the two approaches may complement each other. A distinction into methods employing global properties, edges or region information for segmentation is also common. Besides, there are methods which are particularly suited for color images. In the following, a series of segmentation methods is presented in ascending order of complexity, because this property is especially important in the problem at hand.

2.3.1 Thresholding

A global method of central application in image segmentation is *thresholding*, because it fast, simple in implementation and it can be understood intuitively [25]. In its simplest form, the segmented image is given by the following function:

$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > t \\ 0 & \text{if } f(x,y) \leq t \end{cases} \tag{1}$$

Where $f(x,y)$ is the gray scale input image at location (x,y) and t is a scalar. When t is a constant applicable over the entire image, the equation is referred to as *global thresholding*. Global variants with more than one fixed threshold are possible, too. When t changes over the image, the term *variable thresholding* is used. The term *local thresholding* may be used to point out, that the threshold t at any point (x,y) depends on the properties of a neighborhood of (x,y). A taxonomy of thresholding schemes is presented in Figure 6.

If the intensity distributions of objects and background pixels are sufficiently

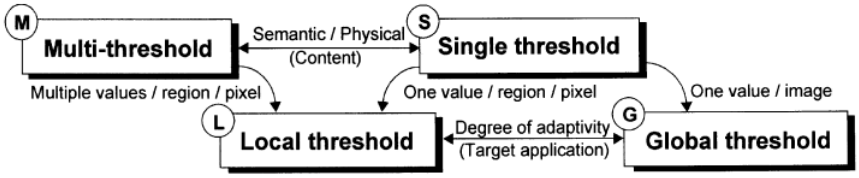

Figure 6: Taxonomy of thresholding schemes (taken from [50])

distinct, a global threshold may be applied. The threshold may be found using an iterative procedure or e.g. with the Method of Otsu [43], which relies on a probabilistic approach. In this method, information from the normalized gray scale histogram is used to find an optimal threshold in the sense of minimizing the intra-class variance. Image averaging or incorporation of knowledge about edges may improve the results for global thresholding. Whenever non-uniform lighting is present in an image, this method will fail. In this case, adaptive thresholding might be an option. Common measures for the determination of the current threshold T_{xy} are the mean m_{xy} and standard deviation s_{xy} within

a local neighborhood. In the Sauvola Method, an adaptive local threshold is computed from the following expression (see the work of Sauvola et al. [50]):

$$T_{xy} = m_{xy}[1 + k(\frac{s_{xy}}{R} - 1)] \tag{2}$$

In the above expression, R and k are constant over the image. Despite giving good results for non-uniform lighting, a major drawback with local methods is the increased computational effort with larger neighborhood sizes.

2.3.2 Edge Detection

Edge based segmentation methods rely on edges found in an image by edge detection operators [57]. Human perception employs similar methods for the recognition of objects. Edges occur at image locations of discontinuity in gray level, color, texture or other features. For grayscale images, they may be obtained by evaluation of the first derivative (*e.g. Sobel Operator*), the second derivative, (e.g. *Lapacian of Gaussian (LoG)*) or the image gradient. Use of higher-order derivatives makes the resulting edge map increasingly susceptible to image noise. After detection, thresholds may be applied to suppress weaker edges. In general, further processing, such as linking of edges is necessary to get a segmentation result. A popular method is the Canny edge detector by J. F. Canny (see [7]). It was designed with the following properties in mind (adapted from [25]).

- Low error rate: All edges should be found and there should be no spurious responses.
- Edge points should be well localized: The located edges must be as close as possible to the true edges.
- Single edge point response: The detector should return only one point for each true edge point. Local maxima around the true edge should be a minimum.

Canny edge detection involves smoothing the image with a Gaussian filter to reduce noise. Then, gradient magnitude and angle images are computed. Non-maxima in the gradient image are suppressed in the next step, followed by thresholding and analysis of connectivity. So, the process calls for definition of a smoothing parameter σ and two thresholds T_L, T_H.

Color edge detection may be carried out by a combination of results on individual channels or by using operators that exhibit the nature of color images, which may be interpreted as a 2D vector field (see [36] for a survey of color edge detection techniques). The detection of edges may serve as an initial step for other segmentation methods, such as the Hough Transform.

2.3.3 Region Growing

Region growing aims for direct construction of regions [57]. This technique starts from a series of pixels, which are called *seeds* and might be found from prior knowledge or randomly. The similarity of regions is used as a growing

criterion, which may e.g. be based upon gray level, texture, color or shape. The definition of the process is as follows:

$$R = \bigcup_{i=1}^{S} R_i, R_i \cap R_j = 0 | i \neq j \tag{3}$$

$$H(R_i) = TRUE, i = 1, 2...S \tag{4}$$

$$H(R_i \cup R_j) = FALSE, i \neq j, R_i \text{ adjacent to } R_j \tag{5}$$

Essentially, the entire image is to be segmented into disjoint, but spatially connected regions that meet a homogeneity criterion $H(R_i)$. The segmentation result depends on the choice of seeds points and possible thresholds for the criterion. Another property that is worth noting, is the need for a stopping criterion.
The obvious approach is to let one region grow, until there are no changes and to proceed with the next seed then. So one region might dominate the growth process. To tackle this problem, *simultaneous region growing* methods may be used.
Region growing is generally better suited to noisy images, where edges are hard to detect. In applications involving mobile robots, color region growing is often used, because objects in this application may be identified by color and and efficient implementation is possible (see e.g. [35]).

Region Splitting and *Region Merging* are related techniques which tackle the problem of segmentation using equal sized image areas. In region splitting, the image is analyzed according to a homogeneity criterion. This possibly partitions the (sub-) image into four quadrangles, until all regions are homogeneous. So, the underlying data structure is a quad tree.
In region merging, small partitions of quadrangular areas are merged according to a similarity criterion. A combined method is called *Region Splitting and Merging* and makes use of pyramid representations.

2.3.4 Mean Shift

A generic non-parametric method for feature space analysis is the *Mean Shift* procedure, which was developed by D. Comaniciu et al. (see [10]). In this method, modes of a density function must be found, given discrete data sampled from that function. For application of the method, the type of kernel for smoothing, the type of features, initial locations and the window size (bandwidth) need to be defined. In an initial step, start locations are placed uniformly on the data. The window is moved in the direction of the computed mean (mean shift vector) for the window. The process will end at a mode of the corresponding density function. After merging of windows that stop at the same mode, the data that these windows have gone through is clustered together. The method can be used for clustering of arbitrary features such as image gradients or even color, which makes it particularly useful for image segmentation. Other fields of use include edge preserving filtering or tracking. The procedure is of considerable complexity and scale sensitive, since the bandwidth must be specified. One solution is to place a small window at each pixel, followed by application of the mean shift and subsequent scaling of the window. The whole operation may be parallelized to a high degree then.

2.3.5 MSER

Maximum Stable Extremal Regions (MSER) were proposed by Matas et al. as a local interest point detector for grayscale images (see [40]). This type of extremal region is defined by an extremal property of the intensity function in the region and on its outer boundary. The process involves subsequent thresholding of the image (*immersion analogy*) and keeping track of the connected regions at each threshold t. The aim is to obtain stable regions, whose support remains largely unchanged over a range of thresholds. So, the algorithmic basis is similar to that of a watershed transform (see e.g. [25] for an introduction on watershed segmentation). The output of the algorithm are positions of local intensity minima (or maxima) along with their corresponding threshold. It is common to maintain additional information about each region which allows a visualization of MSERs as ellipses. Although complexity was already almost linear in the original implementation, this has recently been surpassed by an implementation of linear complexity (see the work of Nister et al. [42]). As an interest point detector, MSERs are invariant to affine transformations in image intensities. Since no smoothing is employed in the process, both fine and large structure can be detected.
An MSER-based method for the segmentation of color blobs was proposed by Donoser et al. (see [14]). The approach requires a one-time initialization of the region of interest and performs color segmentation in LUV space (approximately perceptually uniform) with low computational effort.

2.3.6 Graph Cuts

Segmentation using *graph cuts* is another region-based possibility to partition an image (see the work of Shi et al [56]). The image is represented as a weighted graph $G = (V, E)$, where V denotes vertices and E denotes edges. Each pixel is treated as a vertex and edge weights $w(i, j)$ correspond to pixel similarities. This graph is to be partitioned (by a removal of edges), into disjoint sets $V_1, V_2, ..., V_m$, where by some measure the similarity among the vertices in a set V_i is high, and, across different sets V_i, V_j, is low. The degree of dissimilarity between two cut pieces may be computed as the total weight of removed edges. To reduce bias, a new criteria, the *normalized cut*, takes into account the total edge connections to all nodes in the graph:

$$Ncut(V_i, V_j) = \frac{cut(V_i, V_j)}{assoc(V_i, V)} + \frac{cut(V_i, V_j)}{assoc(V_j, V)} | assoc(V_i, V) = \sum_{u \in v_i, t \in V} w(u, t) \quad (6)$$

In this case, segmentation may be formulated as a recursive algorithm with subsequent solution of an associated eigenvalue problem. The approach is applicable with both grayscale and color images and may be accelerated using multi resolution techniques.

2.4 Determination of Object Size

Geometric properties of pharmaceutical pills are critical for successful identification. For this reason, the rapid identification scheme includes a nonius for accurate size measurements up to 0.1mm (see [28]).
The use of a single image as basis for measurements is supported by the fact

that mobile devices have limited computational resources. In addition, the user should not be overtaxed with the need to take further images, possibly from selected poses. A drawback is that this virtually eliminates the possibility of obtaining measurements in three dimensions.
In general, measurements involve a relation with reference quantities. In certain applications where the scene geometry is fixed and the camera parameters are known, measurements of object properties may be performed without additional knowledge. Unfortunately this is not the case with the problem at hand, so other methods for measuring have to be applied.

2.4.1 Metrology From a Single View

In order to obtain measurement data like object length and width, additional information about the scene is necessary. In [11] a method of *metrology on planes* is presented, which requires computation of a homography between an image plane and a world plane. A homography describes a bijective mapping of image coordinates $\mathbf{x_I}$ to world plane coordinates $\mathbf{x_W}$ and can be represented by a non-singular matrix $\mathbf{H}$ of size 3x3 with 8 degrees of freedom. Using homogeneous coordinates, the relation is as follows:

$$\mathbf{x_{Ii}} = \mathbf{H}\mathbf{x_{Wi}} \tag{7}$$

Such a homography may be computed from a minimum of four point correspondences and does not require knowledge of internal camera parameters (see Figure 7). For n correspondences, $2n$ equations in 8 unknown variables are

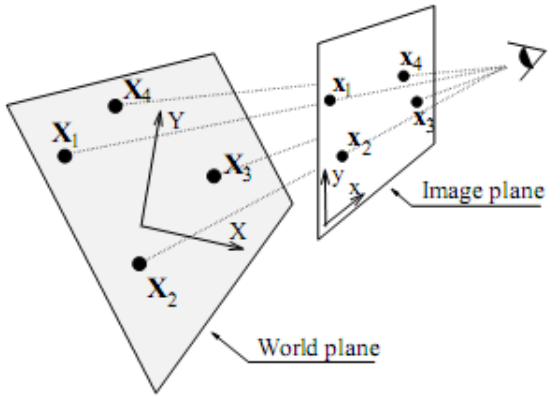

Figure 7: Correspondences for the computation of a plane-to-plane homography (taken from [11])

obtained. A single correspondence $[\mathbf{x_{Ii}}, \mathbf{x_{Wi}}]$ gives the following two equations:

$$\begin{aligned} h_{11}x_{Ii} + h_{12}y_{Ii} + h_{13} &= h_{31}x_{Ii}x_{Wi} + h_{32}y_{Ii}x_{Wi} + h_{33}x_{Wi} \\ h_{21}x_{Ii} + h_{22}y_{Ii} + h_{23} &= h_{31}x_{Ii}y_{Wi} + h_{32}y_{Ii}y_{Wi} + h_{33}y_{Wi} \end{aligned} \tag{8}$$

This system of linear equations may be solved in an exact manner for four correspondences. For a higher amount of correspondences, several methods of estimation exist. A detailed description of the computation and estimation process is given in [27]. With this method, metric measurements within a world

plane are possible. The distances may be computed by evaluating the Euclidean distance between world plane points. In general, the expected accuracy is dependent on

- distortions caused by the optical part of the camera,
- the amount of correspondences,
- the accuracy of correspondences (deviations from their actual location),
- the location of measurement,
- and the accuracy of the estimation technique (for more than four correspondences).

Regarding the problem of recognizing pills, scene information must be made available from the image. Besides determination of these reference points, the exact coordinates of the corresponding world points must be known. With this information, a homography may be estimated and metrology is possible.

This only holds for cameras that provide undistorted images, however. In general, real cameras give non-ideal images, in which the major type of deviation is radial distortion. In this case the projected point is related to the ideal point by a radial displacement. The following definition was taken from [27]:

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = L(\tilde{r}) \begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix}, \tag{9}$$

where

- $[\tilde{x}; \tilde{y}]$ is the ideal image position.
- $[x_d; y_d]$ is the image position after radial distortion.
- $\tilde{r} = \sqrt{\tilde{x}^2 + \tilde{y}^2}$ is the radial distance from the center of radial distortion.
- $L(\tilde{r})$ is a distortion factor, which is a function of the radius $\tilde{r}$ only.

A correction for this type of distortion is possible in pixel coordinates in the following way:

$$\hat{x} = x_c + L(r)(x - x_c) \tag{10}$$

$$\hat{y} = y_c + L(r)(y - y_c), \tag{11}$$

where $[x, y]$ are the measured coordinates, $[\hat{x}, \hat{y}]$ are the corrected coordinates, $[x_c, y_c]$ is the center of radial distortion (often the principal point), with $r^2 = (x - x_c)^2 + (y - y_c)^2$. The function $L(r)$ is only defined for positive values of r and a non-unity aspect ratio needs to be corrected, when computing r. As an arbitrary function, an approximation is possible by a Taylor expansion $L(r) = \kappa_1 r + \kappa_2 r^2 + \kappa_3 r^3 + ...$ Then, the coefficients for radial distortion are κ_1, κ_2 and so on. Center and coefficients for radial distortion may be obtained from a camera calibration step by minimizing a cost based on deviation from a linear mapping. It is possible to correct coordinate pairs, or warp the entire input image using this information, which should improve the accuracy of measurements. The latter may introduce aliasing effects, however.

2.5 Shape Descriptors

Shape is a powerful feature for the task of object recognition and provides the human observer with valuable information. Regarding its shape, an object may be represented in various ways. Among them are chain codes, signatures, distribution of bending energy, chords, skeletons, Fourier transformation of the boundary or polygonal approximation (see [25] and [57]). In literature a large number of descriptors may be found that aim to capture shape properties of an object. There are major differences in basic descriptive power, invariance in regard to transformations (in general, no invariance to perspective distortion), invariance regarding occlusion and complexity for computation and matching. In the context of the problem at hand, a suitable descriptor should be robust to noise (imperfect segmentation), translation, scale and rotation and allow for classification or matching with justifiable effort.

2.5.1 Simple Descriptors

Simple scalar region descriptors (*heuristic region descriptors* [57]) may be seen as a basic approach to characterize objects by their shape, using only the contour of an object. Reasons for the application of these descriptors could be the following [46]:

- They are simple and have general applicability.
- They are semantically simple (similar to human description).
- A combination into a feature vector is possible.
- Some correlation between different descriptors is acceptable (e.g. convexity and compactness).

As shown in Figure 8, features of this kind could be convexity, the ratio of principal axis, compactness, circular variance and elliptic variance. This means that properties of the specific shape are of interest as well as deviations from predefined analytical shapes. In the following, a more detailed description is given.

Convexity is defined as the ratio of the perimeter of the convex hull of an

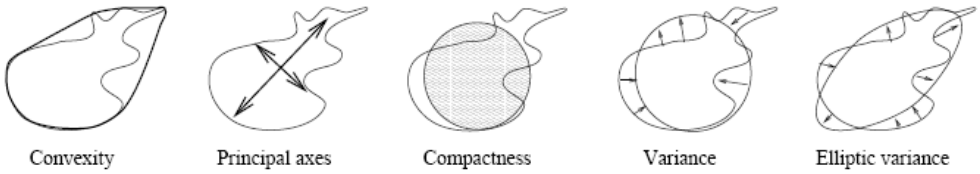

Figure 8: Examples of simple shape descriptors (taken from [46])

object to the perimeter of the original object. The definition is as follows:

$$conv = \frac{p_{convexhull}}{p} \tag{12}$$

This involves computation of the convex hull (e.g. Graham Scan [26], which has a time complexity of $O(NlogN)$, where N is the number of considered points).

Principal axes of an object can be defined as the segments of lines that cross each other orthogonally in the centroid of the object and represent directions with zero cross-correlation. So, a contour is considered an instance of a statistical distribution. Computation involves determination of the covariance matrix $\mathbf{C}$ (see Appendix C). The *ratio of principal axes* as a measure of elongation can be computed in the following way, which saves direct computation of eigenvectors and eigenvalues.

$$prax = \frac{c_{yy} + c_{xx} - \sqrt{(c_{yy} + c_{xx})^2 - 4(c_{xx}c_{yy} - c_{xy}^2)}}{c_{yy} + c_{xx} + \sqrt{(c_{yy} + c_{xx})^2 - 4(c_{xx}c_{yy} - c_{xy}^2)}} \tag{13}$$

The computational effort is $O(N)$, due to the computation of $\mathbf{C}$.

Compactness may be defined as the ratio of the squared perimeter of an object and its area. The minimum (2π) is reached with circular objects. A different definition takes the ratio of the perimeter of a circle with the same area as the object and the original perimeter.

$$comp = \frac{p_{circle}}{p} = \frac{2\sqrt{a\pi}}{p} | a_{circle} = a \tag{14}$$

Area a and contour length p both require a single traversal of the contour, so complexity is $O(N)$.

Deviations from circular or elliptic templates can be measured with *circular variance* (*cvar*) and *elliptic variance* (*evar*). Circular variance is the proportional mean-squared error with respect to a solid circle:

$$cvar = \frac{1}{Nm_r^2} \sum_i (||\mathbf{p_i} - \mathbf{m}|| - \mathbf{m_r})^\mathbf{2}, \tag{15}$$

where $\mathbf{p_i}$ is a contour point, $\mathbf{m}$ is the centroid and m_r is the mean radius. The measure is zero for a perfect circle and requires $O(N)$ calculation steps.
With elliptic variance an ellipse with equal covariance matrix $\mathbf{C_{ellipse}} = \mathbf{C}$ is assumed and the mapping error is measured. Computation is as follows and requires $O(N)$ steps:

$$evar = \frac{1}{Nm_{rC}} \sum_i (\sqrt{(\mathbf{p_i} - \mathbf{m})^\mathbf{T} \mathbf{C^{-1}} (\mathbf{p_i} - \mathbf{m})} - m_{rC})^2, \tag{16}$$

where $m_{rC} = \frac{1}{N} \sum_i \sqrt{(\mathbf{p_i} - \mathbf{m})^\mathbf{T} \mathbf{C^{-1}} (\mathbf{p_i} - \mathbf{m})}$.

Although convexity and compactness may show higher correlation values, the presented descriptors are simple in computation and may give satisfactory recognition results, even for objects of irregular shape.

2.5.2 Image Moments

With image *moments*, an image function is interpreted as a probability density of a 2D random variable (see [57]). There are variants that are invariant to

translation, rotation, scale or even affine transformations. Besides, some may be computed using only the object boundary. Such random variables can be described by statistical characteristics which are called moments and may be used for description of binary or gray level regions. It is worth noting that image moments may be generalized up to arbitrary order. For a digitized image $f(x, y)$ of size MxN, the 2D moment of order $(p + q)$ is defined as (see [25]):

$$m_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} x^p y^q f(x, y), \tag{17}$$

where p and q are integers. The corresponding central moments are invariant to translation:

$$\mu_{pq} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (x - x_c)^p (y - y_c)^q f(x, y), \tag{18}$$

where $x_c = \frac{m_{10}}{m_{00}}$ and $y_c = \frac{m_{01}}{m_{00}}$ are the coordinates of the centroid. Normalized or scaled central moments η_{pq} achieve scale invariance:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \tag{19}$$

where $\gamma = \frac{p+q}{2} + 1$, for $p + q \geq 2$. The most prominent moments are the seven *moment invariants*, which are invariant to translation, rotation and scale (see the work of Hu [30]).

$$\phi_1 = \eta_{20} + \eta_{02} \tag{20}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \tag{21}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{22}$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \tag{23}$$

$$\begin{aligned} \phi_5 = \quad & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{24}$$

$$\begin{aligned} \phi_6 = \quad & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & +4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \end{aligned} \tag{25}$$

$$\begin{aligned} \phi_7 = \quad & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{26}$$

As mentioned before, special moments may be computed using only contour information of the object, which severely reduces computational complexity (see [58]). Different approaches such as Zernike Moments (for an example of application, see the work of Hse [29]) aim to reduce the amount of information redundancy within the traditional geometric moments or have other useful properties such as an improved resistance to noise.

2.5.3 Fourier Descriptors

Use of the Fourier transform of a contour in shape description was introduced in 1971 by C. Zahn and R. Roskies [70] (see e.g. [25] for a short introduction). A series of coordinate pairs $x(k), y(k)$ have to be obtained by traversal of the contour. The boundary is represented as the sequence of coordinates $s(k) = [x(k), y(k)]$. In this approach, each coordinate pair is treated as a complex number (for a contour of length K).

$$s(k) = x(k) + jy(k), \tag{27}$$

where $k = 0, 1, 2, ..K-1$. The boundary is reduced to a series of 1D coefficients and the new representation may be interpreted as a Fourier pair:

$$a(u) = \sum_{k=0}^{K-1} s(k) e^{-j2\pi uk/K} \tag{28}$$

$$s(k) = \frac{1}{K} \sum_{u=0}^{K-1} a(u) e^{j2\pi uk/K} \tag{29}$$

In this representation, lower frequency components account for global shape, while components of higher frequency add parts of more detail. A boundary approximation is possible, if only the first P coefficients are used (frequency cut-off):

$$\hat{s}(k) = \frac{1}{P} \sum_{u=0}^{P-1} a(u) e^{j2\pi uk/P} \tag{30}$$

In general, only a very small amount of coefficients is necessary to capture the major characteristics of a shape. Unfortunately, Fourier descriptors are not insensitive to translation, rotation and scale. Exhibition of certain properties of the Fourier transform and the associated computational effort will be rewarded with invariance regarding the aforementioned transformations, however.

2.5.4 Pairwise Geometric Histogram

The *Pairwise Geometric Histogram* (PGH) is a descriptor with good discriminative power (see e.g. the work of Evans et al. [19]). It is mainly designed for polygonal shapes, but application to non-polygonal types is possible, if a polygonal approximation is computed as an initial step. In the work of Iivarinen et al. [31], irregular shapes are classified from a PGH descriptor which is applied to the output of a key point detector. This application of key points is able to reduce the effort for PGH computation considerably. An efficient detector may be found in the work of Zhu et al. [71]. In this approach a small number of significant points is computed from the input boundary, which are called *critical points*. To obtain these, the boundary must be converted to polar coordinates. Then, several types of critical points are computed by application of criticality metrics for point triples with subsequent removal of non-critical points.

The PGH descriptor makes use of 2D oriented line segments. Their relative orientation and perpendicular distance is analyzed and this information is collected in a 2D histogram. The set of possible angles and distances is mapped

onto the histogram by accumulation of occurrences. During the process, each line is used as a reference line and the angle, as well as the perpendicular distance is computed to all other lines. In fact, every line is represented as a histogram and the accumulation of all histograms represent the object shape. An exemplary histogram is shown in Figure 9. The descriptor has the advantage

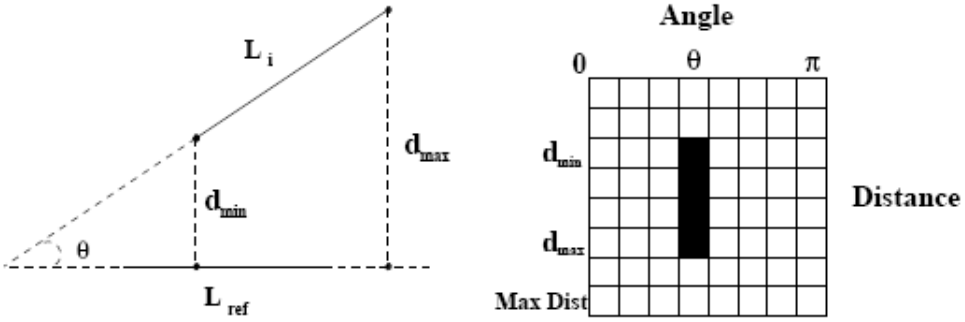

Figure 9: PGH construction of a single line (taken from [31])

of being invariant to starting point, translation as well as rotation. Unfortunately, it is not scale invariant in its original form. Limited scale invariance for object recognition may be achieved by application of a suitable and stable similarity metric (see [2]). A common metric is the Bhattacharyya distance, which is based on the corresponding coefficient. The latter may be calculated for two DxA histograms H_1 and H_2 in the following way:

$$d_{Bh} = \sum_{a=1}^{a=A} \sum_{d=1}^{d=D} \sqrt{H_1(d,a) \cdot H_2(d,a)} \tag{31}$$

When using this metric, shapes that are more similar show a higher value.
In order to limit the computational effort when dealing with non-polygonal shapes, an initial approximation algorithm should aim for a severe reduction of input vertices. Another drawback is the (possibly) high dimensionality of the histogram. This makes subsequent classification steps tedious and would prohibit the application of simple distance measures. A remedy may be found in the computation of conditional expectations of the 2D histogram (see [31]). This reduces the dimensionality of a DxA histogram to $1x(D+A)$. With suitable normalization, the PGH information may be used as a feature vector then.

2.5.5 Shape Context

Shape context is a descriptor for object silhouettes (boundaries which can be represented by a single closed curve) that may be used for measuring shape similarity and recovering point correspondences (see the work of Belongie and Malik [3]). The descriptor is computed from a small number N of sample points (not necessarily key points). Then the vectors originating from such a point (reference point) to all other points are considered. Similar to the PGH approach, the covered angle and distance space (log-polar distances increase resolution for nearby pixels) is mapped onto the histogram using bins. They express the appearance of the entire shape relative to this particular reference

point. If N gets large and the process is repeated with every point as a reference, an exact representation may be obtained, which is called shape context (see Figure 10). Invariance to translation is intrinsic to this descriptor and scale

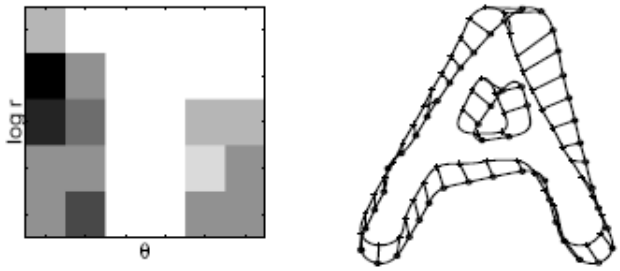

Figure 10: Shape Context (single reference point) for an exemplary boundary (taken from [3])

invariance may be achieved my normalizing all radial distances by the median distance between all point pairs on the shape. For an invariance to rotation, angles can be measured relative to the tangent angle at each point (relative frame). This prohibits incorporation of rotation-variant appearance features (e.g. local orientation), however.
Recognition or matching must be performed using a special similarity metric.

2.6 Determination of Object Color

In the application at hand, one or more color(s) of a pill need to be estimated, where the basic hues of pills (at least by their names) are known beforehand. Provided a suitable segmentation is already available, this is a task of *color classification*. Consequently, this section provides an introduction to color perception. In addition, further considerations on color distance as well as classification are provided, since these topics are important for the subtask of color estimation.

A change in lighting also changes the features of reflected light into a sensor [37]. For a set of surfaces and lighting conditions, human vision allows almost constant perception of a color sensation. An object may be perceived as having the same color in daylight as well as in an environment lit by an artificial light source. This means that the perceived color is not a direct result of the spectral distribution of the received light. This effect is called *color constancy* in color vision and depends on scene complexity and depth. It will be considered in Section 2.6.2.

What man perceives as color, are attributes of the nature of light, as it is reflected by an object (see [25]). Human sensation of a colored (chromatic) light source can be characterized by the following quantities:

- hue: the perceived color (dominant wavelength)
- saturation: relative purity, amount of white light (inversely proportional)
- brightness: achromatic notion of intensity, subjective descriptor

The quantities hue and saturation are referred to as *chromaticity*. On the sensory level, human perception of color is provided by cones, which are photo receptors that differ in their amount as well as sensitivity across the visible spectrum. Color sensations that correspond to maxima in the cone sensitivity curves are called primary colors. In 1931, the CIE (International Commission on Illumination) assigned specific wavelengths to the primary colors, which allows description of colors by color models (for a short introduction see Appendix A).

In vision systems is is desirable to have a meaningful measure for the difference of colors with respect to human perception. This quantitative assessment of color differences will be discussed in the following Section.

2.6.1 Color Difference

The difference between two pixel values in a color space is called color distance (see [37]). The numbers that describe the different color distances in the respective color model in general do no correspond to the color differences of human perception. In the following the quantification of a difference between two instances of color is considered, without accounting for the influence of lighting.

In case of an Euclidean color space (e.g. RGB space), an obvious choice for the calculation of a color difference is the *Euclidean distance*. This distance metric is widely used on account of its simplicity. The Euclidean distance can be computed in RGB space as

$$\Delta E_{RGB} = \sqrt{(\Delta R)^2 + (\Delta G)^2 + (\Delta B)^2}. \tag{32}$$

One unit of this quantity is termed *JND (just noticeable difference)*, as it is the threshold at which a trained observer would notice a difference. It is worth noting that applicability depends on the specific color space and on the amount of perceptual distance of colors. It may poorly resemble human perception of color distance in non-uniform color spaces, such as the RGB space.

More recent equations are calculated in (approximately) uniform color spaces such as CIE LAB and perform weighting of difference components. The following survey of color difference formulas is taken from [65].
CIE LAB and CIE LUV color difference equations from 1976 are widely used today. For CIE LAB, the Euclidean distance is defined as

$$\Delta E^*_{CIE76} = \sqrt{(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2} \tag{33}$$

During the subsequent development of color difference formulas, a dependency on hue has been discussed for brown, purple-blue and green colors. An improvement over CIEDE74, albeit without explicit correction of hue dependency, is the color difference equation CIEDE94.

$$\Delta E_{CIE94} = \sqrt{(\frac{\Delta L^*}{k_L W_L})^2 + (\frac{\Delta a^*}{k_C W_C})^2 + (\frac{\Delta b^*}{k_H W_H})^2} \tag{34}$$

The coefficients W_L,W_C,W_H are obtained from evaluation of weighting functions which depend on material. The coefficients k_L, k_C and k_H can be used to

account for differences from reference viewing conditions.
In an approach to avoid violation of the definition of color difference as a vector distance by introducing correction formulas, a non-linear transform of CIE LAB parameters is used in the DIN99 color difference formula.
The MV-1 color difference formula ΔE_{MV-1} was also presented in 1999 and uses weighting functions for the individual differences to correct a dependency on hue.
Both weighting and space transformation are finally incorporated into the CIEDE2000 color difference formula.

$$\Delta E_{CIE00} = \sqrt{(\frac{\Delta L'}{k_L S_L})^2 + (\frac{\Delta C'_{ab}}{k_C S_C})^2 + (\frac{\Delta H'_{ab}}{k_H S_H})^2 + R_T(\frac{\Delta C'_{ab}}{k_C S_C})(\frac{\Delta H'_{ab}}{k_H S_H})} \quad (35)$$

In an evaluation of color difference metrics using paint samples under a D65 illuminant, the CIEDE2000 formula outperforms all competing difference formulas. Soon after its introduction, CIEDE2000 was recommended by the CIE for industrial color difference evaluation (see the work of Luo et al. [39]).

2.6.2 Chromatic Adaptation

The human visual system may adapt to a large number of viewing conditions. This is also a desirable property in computer vision and it would support generation of invariant features for the problem at hand. Corresponding mechanisms are called *adaptation* (see [55]). They allow to change the sensitivity for a given stimulus depending on the stimulus itself. The human visual system performs three types of adaptation, namely dark, light and chromatic adaptation. The former two differ in speed and allow the system to work across large scales of luminance, whereas the latter is related to the phenomenon of color constancy. Chromatic adaptation is the ability of the human visual system to adjust to changes in the color of illumination. This means that color sensation may appear approximately equal under varying types of illumination. Although models generally reduce chromatic adaptation to the sensory level, this is not entirely feasible. Moreover it is a cognitive process, that involves additional information such as scene complexity.

Models of chromatic adaptation aim to describe the phenomenon on a mathematical basis and may be divided into linear and non-linear approaches. The major steps are estimation of one or more unknown illuminants and subsequent removal of a color cast. The color of the illuminant may be estimated automatically from the image, selected explicitly or obtained by manual examination of a white surface, as is possible with digital still cameras. In the following it is assumed that a suitable estimate for the illuminant color is already available as a triple of values corresponding to scene white.

Chromatic adaptation is often thought to be a result of independent gain control mechanisms on the three types of cone photo receptors in the human visual system (see [55]). This hypothesis lead to the linear *von Kries adaptation model*. In this model CIE XYZ values are transformed into a cone space (ρ, γ, β) and scaled with independent gain control coefficients $\alpha_L, \alpha_M, \alpha_S$, which may be computed

from the maximum cone response in the image. The latter is usually caused by pixels from a white surface. So a von Kries adaptation is often referred to as *white-point normalization*. Transformation to cone space (here: from CIE XYZ) involves a linear transform by a matrix **M**. This model may be expressed as a matrix transform (see Equation 36).

$$\begin{bmatrix} X_{adapted} \\ Y_{adapted} \\ Z_{adapted} \end{bmatrix} = \mathbf{M^{-1}} \begin{bmatrix} 1/L_{white} & 0 & 0 \\ 0 & 1/M_{white} & 0 \\ 0 & 0 & 1/S_{white} \end{bmatrix} \mathbf{M} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (36)$$

Evaluation of the relation produces adapted tristimulus values for a given stimulus. It allows chromatic correction of the image based on knowledge of the current rendering of the color white. A model that can calculate the tristimulus values necessary to obtain a perceptual match across different viewing conditions is known as a *chromatic adaptation transform* (CAT). Such a transform makes it possible to adapt the appearance of an image taken under a specific illuminant to a different one (see Equation 37).

$$\begin{bmatrix} X_2 \\ Y_2 \\ Z_2 \end{bmatrix} = \mathbf{M^{-1}} \begin{bmatrix} L_{w_2} & 0 & 0 \\ 0 & M_{w_2} & 0 \\ 0 & 0 & S_{w_2} \end{bmatrix} \begin{bmatrix} 1/L_{w_1} & 0 & 0 \\ 0 & 1/M_{w_1} & 0 \\ 0 & 0 & 1/S_{w_1} \end{bmatrix} \mathbf{M} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \quad (37)$$

There are different, also non-linear transformations that may be applied in chromatic adaptation. Some of them may be linearized so that application in the von Kries model is possible. A survey on the performance of chromatic adaptation transforms suitable for application in the von Kries model may be found in the work of Süsstrunk et al [60]. Application involves replacement of the Matrix **M** with the corresponding (possibly linearized) version of the model. Bradford, von Kries, Sharp and CMCCAT2000 transformations are shortly introduced and compared on account of their color constancy using metrics for color difference (see Section 2.6.1). Within this survey the original van Kries transform is beaten by the other three transforms by a large margin.

With white balance, the type of color space has a large influence on the result. While the von Kries model calls for balancing in cone space, other spaces have been investigated, some of which may reduce computational effort in an application. A comparison of different options for white balancing with respect to the color space, is given in the work of Viggiano [64]. In addition, results with an approach called illuminant-dependent characterization (see [22]) are also incorporated. Regarding color constancy, the latter performs best, while balancing in native RGB space performs next best. According to this survey, the sRGB space is deemed to be suitable only for correction of very small color deviations, however.
Although white balance changes the colors of an image so that white objects may indeed be perceived as white, is does not fully address the problem of chromatic adaptation. Nonetheless it is a standard feature of digital still cameras nowadays.

2.6.3 Color Classification

In the problem at hand one or more colors of an object need to be identified. This problem is related to that of segmentation, albeit with incorporation of

prior knowledge about image regions in the form of several possible colors. In the following considerations the problem is treated as a color classification task of an image region, based on a series of known hues.

A subproblem in the determination of object color may be seen in the assignment of a color class to a pixel value. This calls for definition of color classes as specific sets of points (point clouds) in a color space. The aim is to generate classes that incorporate a wide range of possible instantiations of a color, may tolerate variable light conditions and allow classification with little computational effort. The following survey of methods for the definition of color classes is based on the work of Kolski [35].

In color classification the geometry of each class or sub-space needs to be defined. Definition is a complex task that involves adjusting a volume to a cloud of samples which were obtained from pixels belonging to objects of the class. The task may be seen as a clustering problem, which may be solved in various ways. One of the simplest techniques is the definition of a pair for thresholds for each channel. In this approach a parallelepiped in a three-dimensional space may be defined by six thresholds. Although this makes class definition rather simple and is very fast in classification, a major drawback is the poor adjustment to the clouds of samples and the high risk of overlapping. Part of the shortcomings of rectangular thresholds may be overcome by transformation into a the so-called TSL chrominance space that is more suited to this type of thresholds (see the work of Dahm et al. [13]).
An alternative are quadric surfaces such as ellipsoids (definable by mean and standard deviations), conics or paraboloids. In general they adapt better to the geometry of a cloud, but make classification slightly more complex.

The used color space should have a considerable influence on the choice of the approximating volume. In RGB space a good choice may be seen in cones, because it will pass by colors of the same chrominance, but varying brightness. A volume approximated by a paraboloid may depict an even better choice, be-

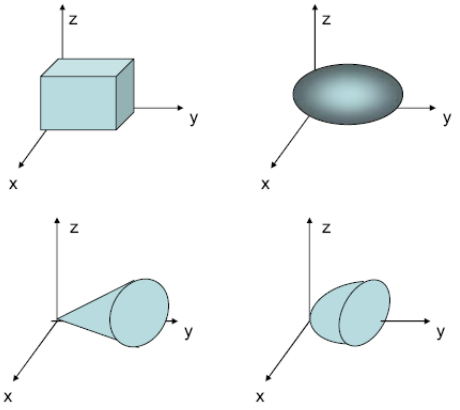

Figure 11: Color subspace geometries: parallelepiped, ellipsoid, conic, paraboloid (taken from [35])

cause it is less susceptible to noise at the origin of the RGB space (see Figure 11 for a visualization of different subspace geometries).
There are several problems that remain even with conic volumes, however. The bounding quality of the surface to the real cloud of samples may be an issue. In addition, the shape of the cloud of samples is influenced by light incidence and reflection. Depending on the color space, the spatial shape of the subspace of interest may not be sufficiently representable by the aforementioned volumes.
Some of the drawbacks may be overcome by using *implicit surfaces* as the bounding volumes. These are 2D geometry shapes that may be defined according to a particular mathematical expression. This implies a function f that defines a surface. When f is below a threshold for a given coordinate pair, it is contained by the surface. The implicit surface is usually characterized by a set of primitives, such as points and lines, that define individual implicit functions. A blending function defines the way in which these primitives will be combined for the final surface.
A major advantage over conics is that implicit surfaces allow easy evaluation whether a sample lies inside a volume. It is also possible to have non-connected parts of a color class after a split of the implicit surface.

A different approach is the application of a clustering technique for the definition of color classes. Scheering and Knoll [51] present a method for image segmentation using a pre-clustered chromaticity plane. In this approach some basic hues need to be defined in HSV space (variant of the HSI space). Different shades of each hue are then added to the basic hue by exhibition of the properties of HSV space and subsequent conversion into YUV space. Finally, a look-up table (LUT) of very small size is constructed by determining the class of the nearest neighbor chromaticity (UV) space. Rather than determination of class membership directly, a look-up table allows classification per pixel in constant time, albeit with additional requirements for storage of the table and at the cost of a loss in classification accuracy due to quantization. A clear advantage is the possibility to represent arbitrarily-shaped regions or volumes within a LUT.

Whenever similar (regarding distance in color space) color classes need to be constructed, point-based mapping of pixel values to classes becomes an ambiguous problem. This overlapping problem may be mitigated by suitable preprocessing (e.g. smoothing) or by deferring the decision. A classification may be possible by investigating the neighborhood of a pixel when no decision was possible at the time of class creation (see the work of Palma-Amestoy et al. [44]). Application of LUT based color classification to an image may give a rough segmentation with the advantage of little computational cost. In color region growing such a classification may be used as part of the homogeneity criterion (see the work of Kolski [35]).

2.7 Classification

2.7.1 Preface

The result of object descriptors is often available as a vector or may be converted into this representation (see Section 2.5). The task of a classifier is to assign a

class label to such an instance. To achieve this, training is necessary in general. Depending on the nature of the problem, different learning approaches may be employed (see the work of Duda et al. [16]). Knowledge of labeled instances would call for an approach using supervised learning, which seems feasible in the current problem, because the amount of samples may be considered small. In this learning method, a series of feature vectors along with their corresponding class labels form the basis for training and application of the classifier.
The aim is to achieve a good recognition rate on the training set, but also good generalization for unknown examples (e.g. test set). In general, a perfect fit to training data is an indicator of over fitting, which may be the cause for a degradation in classification performance on non-training data.

The task at hand calls for at least one application of a classifier. For the current problem, the amount of available samples can be considered small and storage as well as processing capacities are limited, however. With the aforementioned constraints in mind, a list of possible classifiers will be provided, which is based on the work of Kotsiantis [38]. Finally, a series of metrics for the evaluation of classifier performance is given.

2.7.2 Nearest Neighbor

Nearest neighbor is a lazy learning approach, where training involves just storing a series of example vectors. In the classification step the distance to all examples is computed. An unknown example is assigned a class depending on the first k stored examples with minimum distance (k Nearest Neighbor). Often the Euclidean distance is used, but others, like the normalized Euclidean distance (takes into account the variance of each feature) or the Mahalanobis distance (takes into account the covariances between different features) may be employed instead, but this depends on the specific nature of the problem. Given a suitable set of example vectors, the approach calls for the parameter k to be chosen adequately, e.g. by a cross-validation technique (see e.g. [16] for an introduction). Depending on the number of training samples and the specific implementation, storage requirements and classification time may be high. The latter can be considerably reduced by application of a suitable data structure, however.

2.7.3 Naive Bayes

The *Naive Bayes* classifier is based on the Bayes theorem for the computation of conditional probabilities. This approach makes the (naive) assumption that different features were truly independent of each other. In the training step, conditional probabilities need to be estimated, which is often simplified by assuming normally distributed data. Classification of an example involves application of a decision rule, such as MAP (maximum a posteriori) (see e.g. the work of Duda et al. [16]). The Naive Bayes classifier has the advantage of fast training and classification with little memory requirements. One has to keep in mind, that the underlying independence assumption is almost always violated, however. Numerical features may also be used with this classifier, but require a discretization step.

2.7.4 Decision Trees

Decision trees classify instances by sorting them based on feature values. Training involves construction of the tree, usually by application of heuristics (splitting criterion). At every node, a split is carried out that separates the data, which corresponds to a decision based on a specific attribute. The goal at each node is to find a split that optimally separates the data. Over fitting may be addressed by pruning the tree during construction (pre-pruning) of afterwards (post-pruning). An extension to continuous attribute values is possible with the popular C4.5 algorithm by J. R. Quinlan [48]. In classification, the instance vector is passed down the tree and is assigned a label at the resulting leaf. Thus, classification is generally fast and storage requirements are relatively low.

2.7.5 Performance Metrics

The most obvious characteristic of classifier performance is the relative amount of correctly classified examples with respect to the amount of examples (recognition rate). This is not sufficient to fully characterize a classifier, however. Is is also desirable to gather information about misclassification. Such information may be valuable for the purpose of optimization, too. In general, there are four possible outcomes of classification (see e.g. the work of Fawcett [20]):

- True positive: A positive instance is classified as positive.
- False negative: A positive instance is classified as negative.
- True negative: A negative instance is classified as negative.
- False positive: A negative instance is classified as positive.

For a test set, the classification result may be visualized as a so-called confusion matrix (contingency table). This is a square matrix, where the classified labels are contrasted to the real outcomes. Each row and each column denotes a class and entries on the main diagonal correspond to correctly classified results (true positives). This allows a quick evaluation of a classification result.

Several metrics for classifier performance use values computed from this information. Amongst others, a *Receiver Operating Characteristics* (ROC) plot may be created. This 2D plot originates from signal detection theory and is widely used in the visualization and characterization of classifier performance. A ROC may be created by plotting the true positive rate (*tp-rate*, hit rate, recall, sensitivity) against the false positive rate (*fp-rate*, false alarm rate).

$$tp - rate \approx \frac{\#(\text{true positives})}{\#(\text{total positives})} \tag{38}$$

$$fp - rate \approx \frac{\#(\text{false positives})}{\#(\text{total negatives})} \tag{39}$$

Alternatively, the *fp-rate* may be computed as

$$fp - rate = 1 - specificity, \tag{40}$$

where specificity is defined as

$$specificity \approx \frac{\#(\text{true negatives})}{\#(\text{total negatives})} \tag{41}$$

For discrete classifiers, a classification result on a test set corresponds to a single point in the ROC plot. It must be noted that the ROC is insensitive to changes in class distribution, which may be useful when comparing different test sets.

3 Robust Segmentation

3.1 Preface

A suitable segmentation method in the context of a mobile application should be sufficiently robust, but also efficiently computable. On the one hand, it is desirable to find a solution with little restrictions in image acquisition, because this would constrain acceptance of the system. On the other hand, a lack of robustness would lead to the same problem.
In the system to be developed, robust estimation of shape, color and metric properties must be performed. So, it is necessary to establish point correspondences from the single acquired image (see Section 2.4.1), because the setup is not fixed. This can be solved by using a target of known geometry. Although the system may not work without such a target then, this approach has the following advantages:

- Full control over the background adds robustness in segmentation.
- The known target geometry may be used to adjust parameters for segmentation and to reduce the amount of input data.
- Correction of lighting effects becomes possible by using reference measurements.
- Feedback about the expected estimation quality may be given, even before the actual computation.

When using a frame marker, the pills may be positioned within the enclosed area. This choice has the advantage that existing functionality of Studierstube ES can be used in the detection process. A suitable background must be chosen so that segmentation of (in principal) arbitrarily colored pharmaceutical pills is possible, however.
In the following, the subproblem of segmentation will be treated under a series of assumptions.
Image acquisition takes place under adequate lighting conditions and with a variable pose. The pill is entirely captured within the image, of convex shape, considered planar and may have one or two colors. Brick lines as well as imprints may be visible on the surface.

3.2 Image Acquisition

Image regions that correspond to pharmaceutical pills should be sharply defined and be of sufficient size concerning pixel area. It seems desirable to represent objects as big as possible, because, intuitively, this increases the accuracy of segmentation and feature estimation. From the point of computational efficiency, however, it is the exact opposite. Obviously, images of higher resolution require more computational effort as well as memory.
Research in the field shows that data concerning sensor properties for web cams and mobile devices is sparse. In some cases, it is possible to obtain such data from additional information in the image file (EXIF - Exchangeable Image File Format). Consequently, tests with available web cams and mobile phones were carried out. According to these tests, a fixed focus camera cannot be used in

the application at hand, because objects in the image are either too small or not sharp enough. Devices with automatic adjustment of focal length (autofocus) are able to produce satisfying results, but the exact settings are often not known.

3.3 Evaluation of Segmentation Methods

A number of segmentation methods were evaluated, in order to get an impression of quality as well as runtime. For all subsequent experiments, evaluation and exemplary implementations were carried out in Matlab[21]. Corresponding C++ implementations were used, if available. A standard dual-core laptop with a clock speed of 1.8 GHz and 2.5 GB of main memory served as evaluation platform.

In Figure 12 an image of pills within a frame marker with white background is shown. The image comprises shadows and highlights on pills, as is often the case in a non-fixed setup.

Figure 12: Pills on a frame marker with white background

In Table 4, preliminary execution times for various segmentation methods on the image in Figure 12 are shown. In Figure 13, the results of various segmentation methods are presented. The most informative representation in a visual sense, was chosen in each case. In case of Maximum Stable Extremal Regions (see Section 3), this means that regions are represented by ellipses, although the actual region could be obtained from the corresponding threshold.

[21] `www.mathworks.com`
[22] `http://www.vlfeat.org/~vedaldi/code/mser.html`

Method	Implementation	Time [ms]
MSER (A)	C++ (taken from [22])	556
Color Graph Cuts (B)	C++ (taken from [21])	1638
Color Canny Edge (C)	Matlab (implemented after [36])	3667
Mean Shift (D)	C++ (taken from [9])	37785

Table 4: Runtimes for segmentation (640 x 480 pixels, measured in Matlab): image from Figure 12

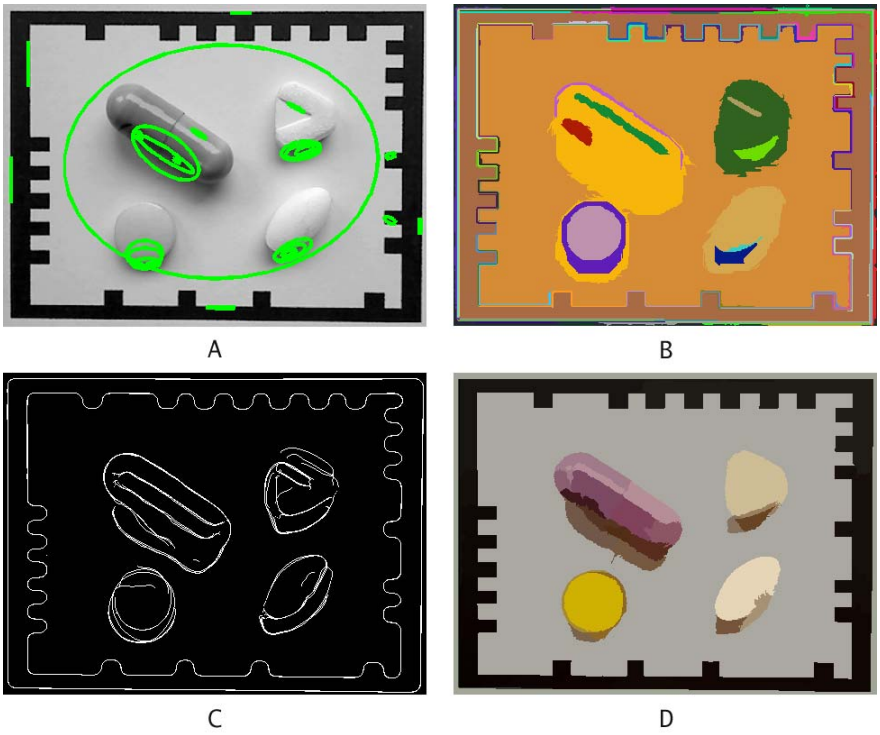

Figure 13: Segmentation results with image from Figure 12 (extract): A...MSER; B...Graph Cuts; C...Color Canny; D...Mean Shift

From visual inspection it is evident that shadows pose a major problem for any of the tested methods. Choosing another background color would not change matters significantly, because pills may have arbitrary colors.
Although the image acquisition device was configured to perform automatic white balance, the colors do not fully correspond to those of the real samples. Visual inspection reveals that particularly brighter pills have apparent deviations in the test image. As a result, a faithful estimation of color will be difficult. The type of deviation may vary according to the light source, however.
It must also be noted that these tests were carried out using an image taken at suitable lighting. Since a real system will be used at varying conditions, the quality of segmentation may be expected to degrade further. When comparing these results and considering the associated run times, it is evident that even the qualitatively better methods perform unfeasibly on uniform background.

3.4 Segmentation on Structured Background

Instead of a homogeneous background, a pattern may be used which allows processing by methods with low computational complexity. Since thresholding is an efficient operation that relies on differences in contrast, a chessboard pattern using the colors black and white is chosen (see Figure 14).

Figure 14: Pills on a frame marker with structured background (square length 0.6 mm, 640 x 480 pixels)

Under the assumption that such a pattern does not occur on a pharmaceutical pill, a segmentation can be obtained by local adaptive thresholding and morphological operations on a grayscale input image (see e.g. [25] for an introduction on morphological operations). All relevant computations may be carried out on a square neighborhood that extends into both directions at least w_h pixels, giving a minimum total square length of $2 \cdot w_h + 1$ pixels. The principal algorithm is as follows:

$$M_{seg} = (\neg(M_{Th_1} \bullet SE_1) \circ SE_2) \bullet SE_1, \tag{42}$$

where M_{Th_1} denotes a mask obtained by local adaptive thresholding with a neighborhood size of $2 \cdot w_h + 1$, and SE_1 and SE_2 denote structuring elements of length $2 \cdot w_h + 1$ and $2 \cdot w_h + 3$. The symbol $\neg$ is used for mathematical inversion and the symbols $\circ$ and $\bullet$ denote morphological opening and closing.
A drawback is the appearance of jagged edges, which hampers the subsequent determination of object shape. Besides, certain patterns of high contrast, as caused by imprints, may lead to artifacts.
Due to the fact that pills are considered to be of convex shape [23], we apply the following procedure to extract the pills on the previously described target:

[23] Querying the Identa database reveals that the amount of concave pills is negligible.

- Segmentation using the approach described above.
- Region labeling and computation of boundaries.
- Computation of the convex hull (see e.g. the work of Graham [26]).
- Linking of points by an interpolation algorithm into connected chains.

With tis procedure a series of smooth region boundaries can be obtained. If necessary, the region itself may be calculated in a flood fill operation then (see [57]). In order to be able to test the approach concerning performance and quality, an exemplary implementation is carried out.

3.4.1 Preliminary Tests and Considerations

From the steps outlined above, region labeling and boundary computation, determination of the convex hull and the flood fill operation can be implemented using already available functions of the test environment. In the face of an efficient implementation, local adaptive thresholding was implemented using integral images (see the work of Shafait et al. [53]). This has the desirable effect that runtime is not dependent on the window size.
As a solution to the interpolation problem of the points from the convex hull, the Bresenham line drawing algorithm was chosen for reasons of efficiency as well as simplicity (see the work of Bresenham [5]).

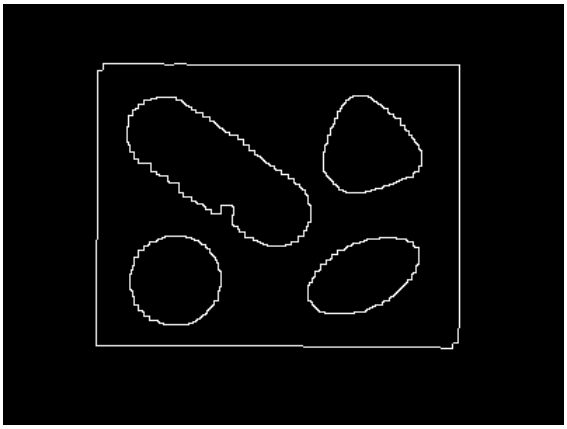

Figure 15: Segmentation borders for Figure 14 (extract)

In Figure 15 a segmentation result of the initial algorithm is shown. As reasoned before, the example shows considerable artifacts due to jagged edges and text on the pills. In Figure 16 the result of the improved algorithm is presented. From visual inspection it is evident, that segmentation quality is notably better,

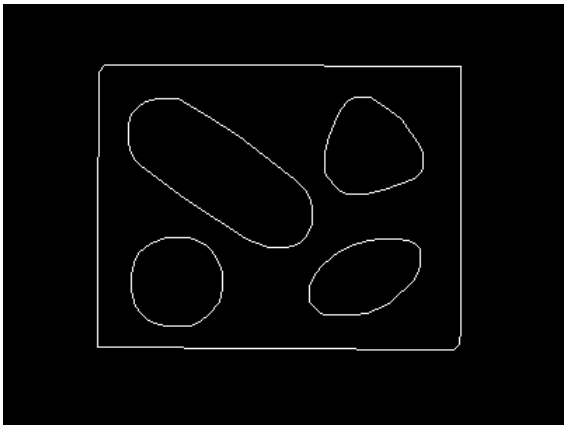

Figure 16: Improved segmentation borders for Figure 14 (extract)

provided that a region has not been split. The variety of conditions concerning image acquisition that the system will have to face, make a more detailed evaluation mandatory.

In most algorithms for image segmentation no assumptions about position and alignment of objects are made, whereas object scale is more critical, because it may have an influence on the segmentation parameters. In the developed algorithm, the scale of objects that may be segmented is determined by the square size of the chessboard pattern, which also determines the window parameter w_h up to a certain degree. At fixed scale, a bigger window size may add robustness, but in general degrades fineness.
A series of tests was carried out to investigate the dependence on translation, rotation, scale, perspective distortion and non-uniform lighting (see Figure 17, Figure 18 and Figure 19).

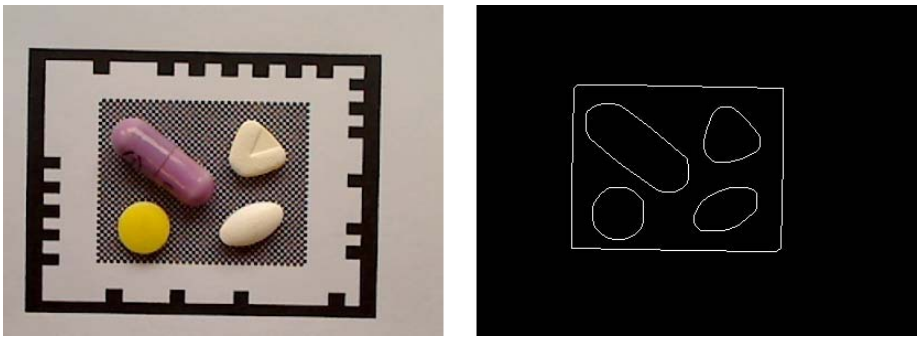

Figure 17: Scale evaluation (from left to right): input image (extract), segmentation result

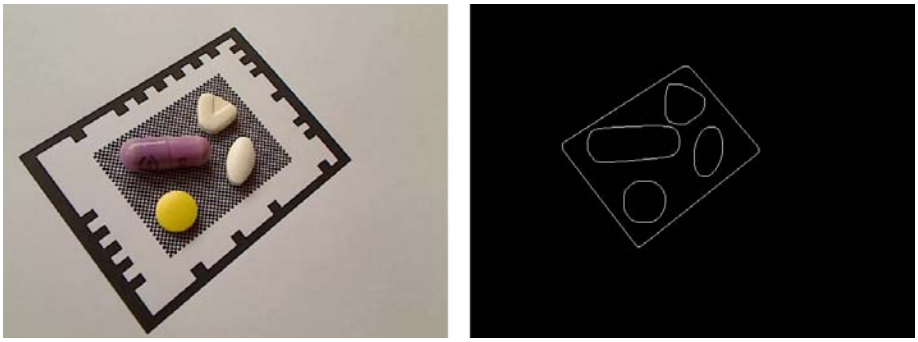

Figure 18: Evaluation with rotation and perspective distortion (from left to right): input image (extract), segmentation result

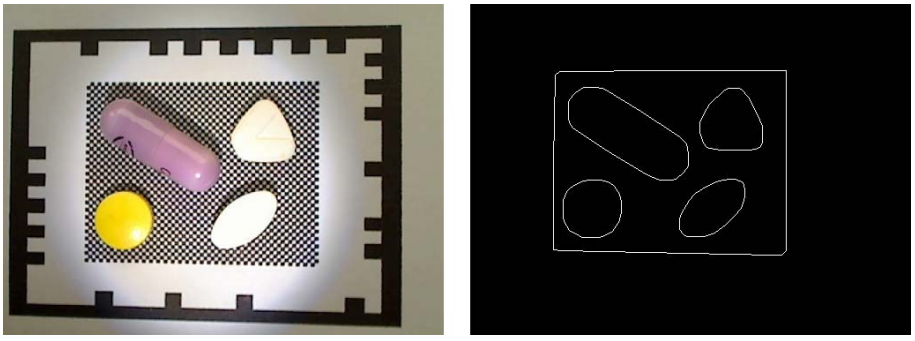

Figure 19: Evaluation with non-uniform lighting (from left to right): input image (extract), segmentation result

It is interesting to note that the window parameter w_h was kept constant during these tests. Although an influence concerning rotation is present, the impact stays confined.
According to the obtained results, the presented algorithm offers sufficient segmentation performance under the conditions that such a system may be expected to operate.

4 Extraction of Object Properties

4.1 Preface

While the estimation of geometric properties provides little margin for subjectiveness, the situation concerning shape and color is different. Unfortunately, no reference data is accessible for the construction of feature estimators in the current application. So, a series of representative examples was obtained from online databases, literature and physical samples.
The context of a mobile application, the proposed segmentation method (see Section 3) and the aforementioned observations lead to the following basic requirements for feature estimators.

- The estimator may be trained from a small number of examples.
- An estimation is possible with justifiable computational complexity and storage requirements.
- The results correspond to human perception.
- The estimator is sufficiently robust concerning the environment of image acquisition.
- An adaptation to new examples is possible with little effort.

For individual estimators, additional properties are desirable, which will be discussed in the corresponding subsections. Where possible and reasonable, an estimator was chosen to consist of a feature extraction and subsequent classification step, which is desirable for reasons of modularity. In addition, this gives potential for further optimization. The following considerations are based on the assumption that the starting position for property estimation is the boundary of an image region that resembles the silhouette of an object as closely as possible. In other words, for the following stage the problem of segmentation is deemed to be solved.

4.2 Shape Estimation

For reasons of efficiency, the boundary of an object was chosen to serve as a basis for the estimation of object shape. Obviously the shape of the boundary is dependent on the pose, because of perspective distortion (see for example Figure 18). This effect can be accounted for by using an available homography or by providing information to the user. Although both approaches are possible in the application at hand, none is actually desirable, because correction would lead to additional computational effort and motivating the user to take a second image would harm acceptance of the system. So, a shape estimator should be as invariant to changes in translation, rotation and scale as possible. As reasoned earlier, a certain degree of perspective distortion should still be tolerable.

4.2.1 Shape Classes

The basis for feature extraction and classification of shape may be established by collecting a series of representative example shapes for each class. The examples for the problem at hand were created in semi-automatic form from images

of physical samples, images obtained from database queries, as well as images found in literature. An analysis of these examples led to the decision to categorize each example into one of 9 representative shape classes. The shape classes represent **convex** *circular*, *oval*, *oblong*, *triangular*, *quadratic*, *rhombic*, *pentagonal*, *hexagonal* and *octagonal* shapes.

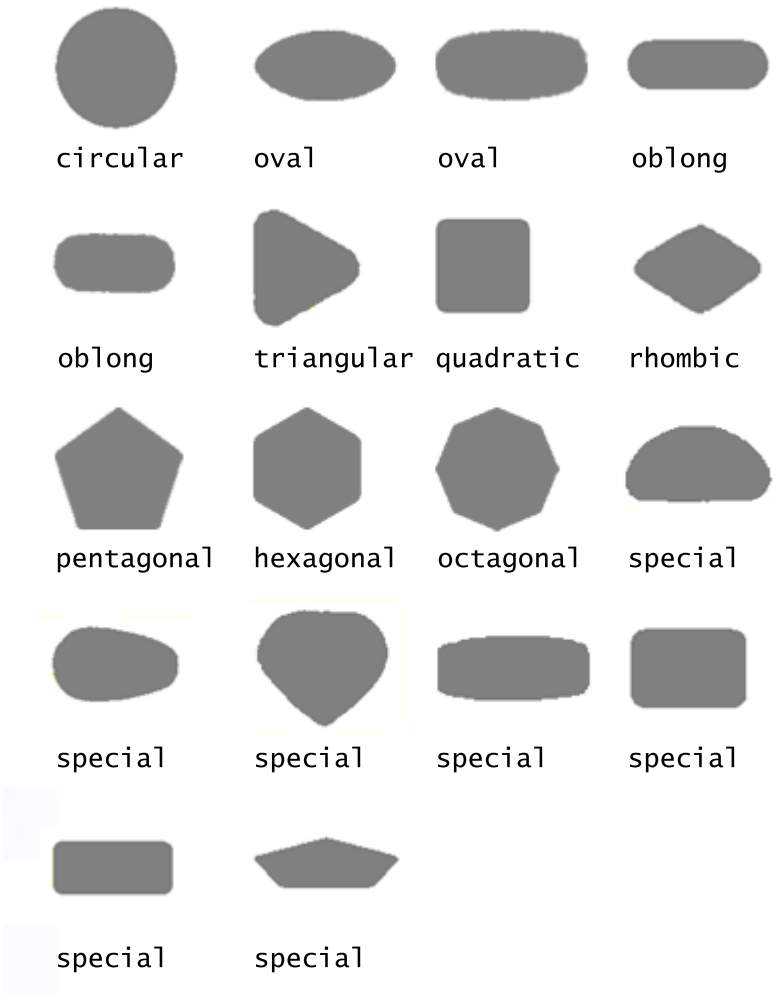

Figure 20: Examples of pill shapes with classes

A class termed *special* was added to represent shapes that do not fall into any of the other categories (see Figure 20). This categorization forms the basis for estimation of shape with supervised learning from labeled examples. Due to the nature of pharmaceutical pills and for reasons of efficiency, shape estimation was decided to be based solely on analysis of the object boundary. So, a rectangular region containing the example was cut from each image and the object as well as the background were dyed so that the boundary could be obtained in a subsequent segmentation step without difficulties.

4.2.2 Shape Features

Considering the investigated shape characteristics of pharmaceutical pills and their pronounced intra-class variance as well as partially low inter-class variance, applicable shape descriptors need to have considerable discriminative power. In addition, the non-fixed setup concerning image acquisition as well as limited computational resources call for descriptors that are largely invariant concerning translation, rotation and scale, but also efficiently computable and classifiable. When considering the shape descriptors of Section 2.5, the following conclusions may be drawn:
Simple shape descriptors and image moments are efficiently computable, but their discriminative power has to be investigated. Their invariance allows application of several classifiers.
Fourier descriptors require additional computational effort to achieve the desired degree of invariance.
The Pairwise Geometric Histogram (PGH) and the Shape Context descriptor are known to have good discriminative power, but require a number of representative points to be extracted from the boundary. Besides, the PGH is not scale invariant in its original form, but neither is the Shape Context.
For the Shape Context descriptor, also rotational variance has to be accounted for in an additional step. Examples may be classified by matching a bipartite graph. This operation has a complexity of $O(N^3)$, where N is the number of key points per candidate (see the work of Belongie and Malik [3]).
In the PGH approach, matching of two-dimensional histograms is necessary, which can be converted to feature vectors so that other classifiers may be applied, however.

Treatment of shape information as a feature vector allows for individual selection and optimization of the classifier. Due to this desirable property, **simple shape descriptors**, **moment invariants** and the **PGH** are applied as shape features during evaluation of shape estimation. While simple descriptors and moment invariants can be implemented in a straightforward manner, the PGH approach requires the solution of an efficiency problem and a scale variance problem.
In order to test the PGH approach, an efficient key point detector is used (see the work of Zhu and Chirlian [71]). Figure 21 shows the output of our algorithm, when applied to an exemplary boundary.

For the boundary in Figure 21, the number of input points for subsequent computation of the PGH can be reduced by more than 90 percent. As only convex shapes are covered in this work, it may be possible to use the points that make up the convex hull as key points. This has to be investigated, however. Further tests showed that this key point detector is able to give satisfying results for all available training and test shapes.
The PGH may then be computed from the key points by creating a two-dimensional histogram with a predefined amount of distance bins D and angle bins A in an $O(N^2)$ operation. In the original approach (see Section 2.5), the projected length on the reference line and the angle affect a series of histogram bins, whose counter is incremented by one. So, a maximum distance has to be

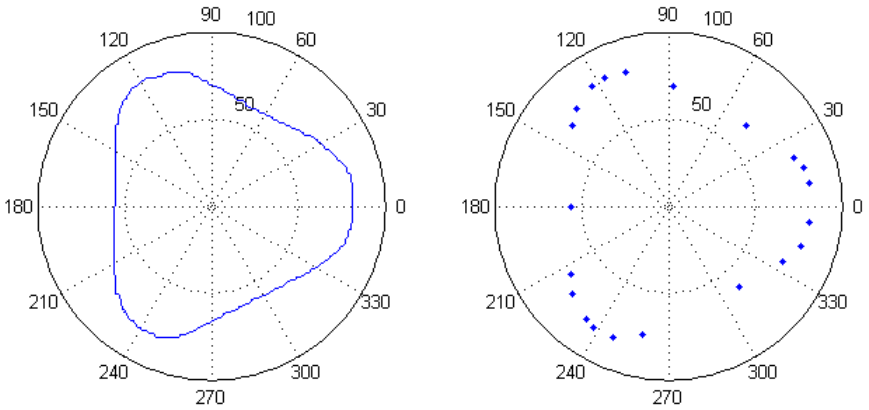

Figure 21: Key point detection on an exemplary shape (from left to right): polar plot of the original boundary, polar plot of critical points

defined before computation, which corresponds to the whole range of distance bins. If a static value is defined and differing scales must be processed, the discriminative power of the descriptor diminishes. Although scale invariance, at least for a limited range of scales, may be achieved by a suitable similarity metric for histogram comparison, this is not possible when a feature vector must be created. So, the maximum distance for a PGH must be obtained from the input shape. Due to the fact that computation of the convex hull is part of the segmentation algorithm, no outliers may be expected. So, a measure of scale may be found just by searching for the maximum distance d_{max} in the key points from their centroid. The maximum PGH distance may be computed as

$$d_{PGH} = 2 \cdot d_{max} \tag{43}$$

then. A feature vector for this *compressed* PGH may be obtained by computation of conditional probabilities on angle bins and distance bins in the histogram and subsequent normalization. This dimensional reduction of the 2D histogram is based on the following operation (computed for distance d, see the work of Iivarinen et al. [31]):

$$E(d) = \frac{\sum_a a \cdot p(d,a)}{\sum_a p(d,a)}, \tag{44}$$

where $p(d,a)$ corresponds to an entry in the 2D histogram at distance d and angle a. In this equation, a iterates through all angle bins.
The values of $E(a)$ may be computed correspondingly (see Figure 22).

All obtained probabilities may then be combined into a feature vector of size $D + A$.
An additional possibility is a classification of the non compressed 2D histogram, which was added as another descriptor during evaluation. In the context of mobile devices, the similarity measure has to be chosen with care, because it can render the problem intractable concerning computational complexity.

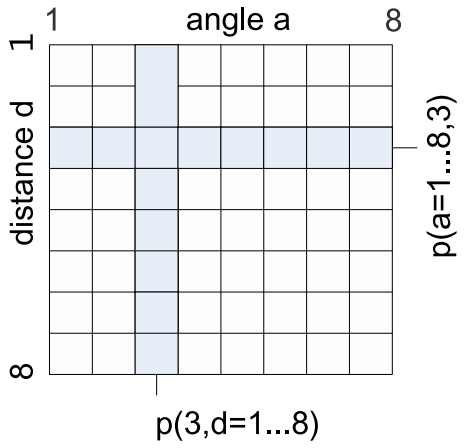

Figure 22: Computation of conditional probabilities (PGH with 8 distance bins and 8 angle bins)

4.2.3 Shape Classification

The key requirements for shape classifiers in the current context are a good recognition rate and fast classification. The effort for training is less critical, because it can be performed in an off line step. From preceding considerations it is evident that the amount of samples for each class is very small and that the dimensionality of the feature vectors of the selected shape descriptors is rather different.

Preliminary experiments were carried out using the *Waikato Environment for Knowledge Analysis (WEKA)* (see [68]). Matlab served as a means of getting shape feature data into WEKA (creation of Attribute-Relation File Format (ARFF) files). In addition, Matlab was used for further testing and subsequent implementations. All tests were carried out using a set of examples for the 10 classes, with a ratio r_{tr} of training examples with respect to test examples of 0.36.

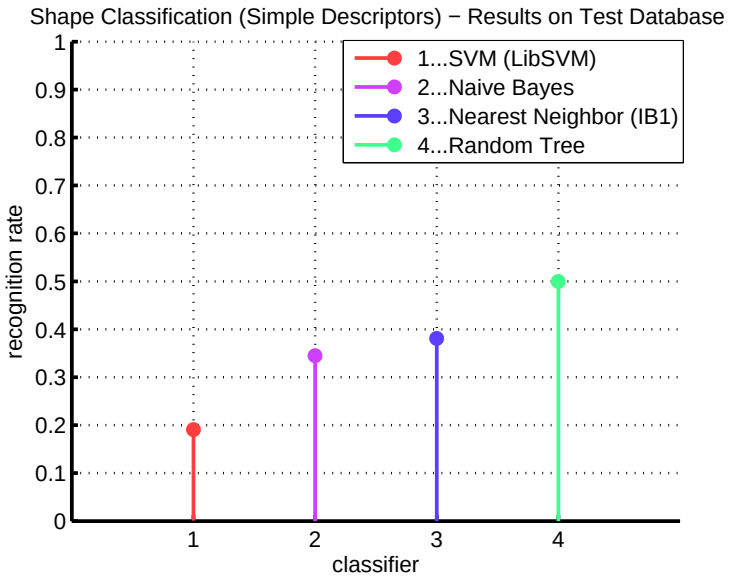

Figure 23: Results of shape classification on test data (10 classes, $r_{tr} = 0.36$): Simple Descriptors

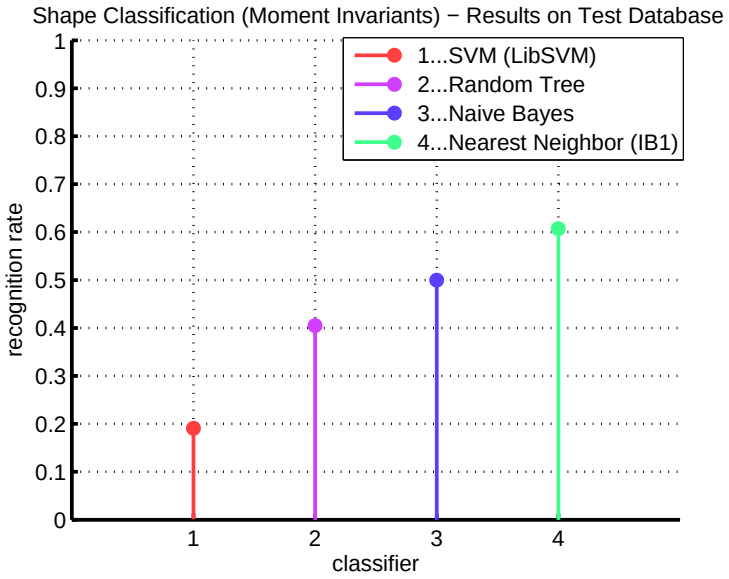

Figure 24: Results of shape classification on test data (10 classes, $r_{tr} = 0.36$): Moment Invariants

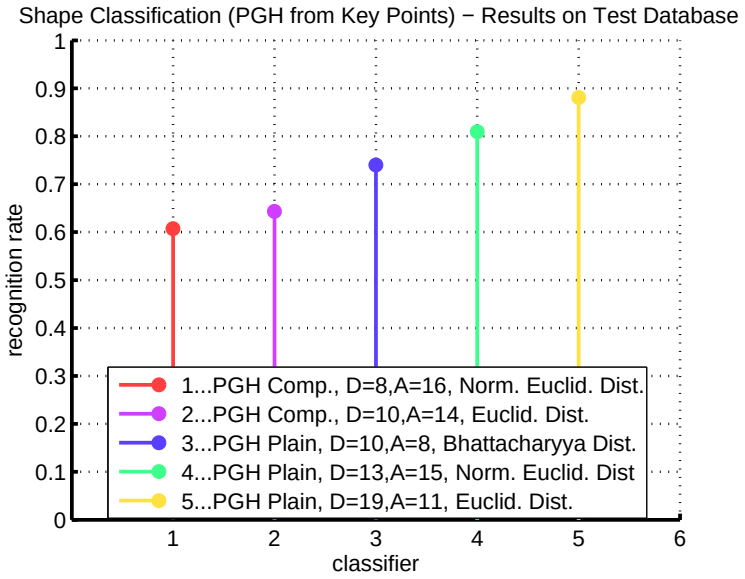

Figure 25: Results of shape classification on test data (10 classes, $r_{tr} = 0.36$): PGH from critical points

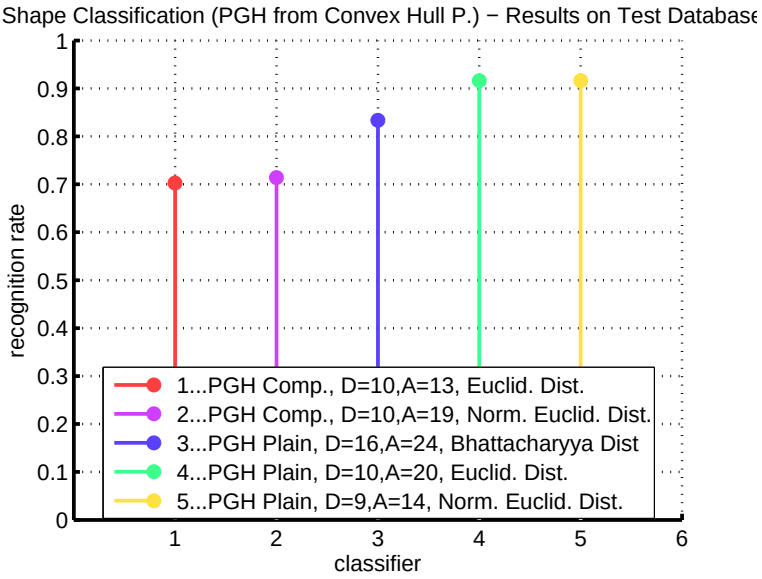

Figure 26: Results of shape classification on test data (10 classes, $r_{tr} = 0.36$): PGH from convex hull points

The obtained results are shown in Figures 23, 24, 25 and 26. For the PGH results, the lowest number of bins for the stated recognition rate is provided. The following conclusions may be drawn:

- Both moment invariants and simple descriptors cannot be used to categorize pill shapes into the selected categories. Moment invariants perform better, however.
- The compressed Pairwise Geometric Histogram is not able to deliver satisfiable performance either, when using an instance based learner with the Euclidean distance measure. The normalized Euclidean distance improves performance, however.
- When using plain PGH data in such an approach, performance increases significantly, but is dependent on the similarity metric again. Interestingly, the Bhattacharyya coefficient cannot reach the performance of Euclidean metrics, despite its greater complexity. The normalized Euclidean distance does not perform notably better than the Euclidean distance, although the added knowledge (variance) should provide better results in general.
- When using the convex hull for key points and subsequent computation of the non-compressed PGH, performance increases over the results obtained by a dedicated key point detector. This may be attributed to the fact that the obtained points describe a less noisy version of the shape.

The behavior of the implemented PGH descriptor concerning scale, deserves additional attention. In order to test the recognition rate with different scales, a test set with scaled circular, octagonal, oval and oblong shapes was created. The lower limit was set at $s = 0.2$ of the original size, and $s = 2.0$ was set as the upper limit, with a step size of $\Delta s = 0.2$. The tests were carried out with a fixed amount of angle bins ($A = 16$) and a variable amount of distance bins ($D = 8...24$), using a nearest neighbor classifier based on the Euclidean distance. In Figure 27, the results with respect to the amount of distance bins are shown.

Although the characteristics are not monotonously increasing (recognition rate varies between 0.81 and 0.94%), there is a slight overall enhancement with an increasing amount of distance bins. So, the PGH approach may give viable results when examples of a single scale are stored. Nonetheless an extension of the training database may be desirable.

The plain PGH approach combined with instance-based classification, requires matching high-dimensional data with several examples, which may render the problem intractable in reasonable time. Nearest neighbor classification can be made very efficient by application of an efficient data structure however. Then, classification time is dependent on the number of PGH dimensions, instead of the amount of training examples. This means that the problem may well be solved with limited computational resources.

Although no perfect classification is possible, the obtained results show that

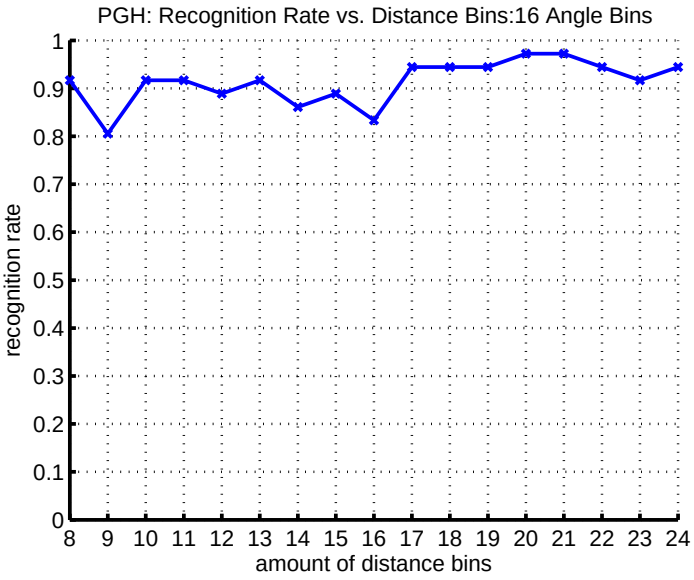

Figure 27: PGH scale test for circular, octagonal, oval, oblong shapes: fixed number of angle bins ($A = 16$), Euclidean distance

the proposed method for shape classification may be used for mobile applications, since it has enough descriptive power and can be efficiently computed using an appropriate data structure. As popular on-line databases use only a few shape classes, better results may be expected (see Section 5.2.2).

4.3 Size Estimation

Geometric properties of pharmaceutical pills such as length, width and height are an important feature for identification. In the rapid identification scheme (see [28]), the length and width of a pill need to be determined. On-line databases such as Identa[24] also offer the possibility to enter a height value in order to improve the results of a query.
It is also important to note that the terms used for length and width may vary according to the type of pill. In the following considerations, the length of a pill is defined as the extension of its convex boundary along the major axis. This suggests that the extension in the rectangular direction is defined as width. With this definition the majority of pill shapes may be measured correctly.
In the determination of these parameters it is also reasonable to use information about the shape. For circular shapes as well as regular polygonal shapes, length and width are (nearly) the same, which is why the average of width and length can be computed as an estimate of their extension.
As with any other feature estimator in the current context, robustness and efficiency are required properties for the estimation of object size.

[24] `http://www.gelbe-liste.de/pharmindex/identa`

4.3.1 Rectification

With a single input image, measurements can be carried out in two dimensions by estimation of a homography $\mathbf{H_{IW}}$ between points on the image plane and points lying on a plane in world space (see Section 2.4.1). It is clear that this only holds under the assumption of negligible distortions from the system of lenses during image acquisition, however.
Using the homography $\mathbf{H_{IW}}$, it is possible to judge the image geometry by examination of angles between imaginary lines that connect consecutive points. Such points, if represented by simple objects or a marker, can be found with little computational effort. Besides, feedback about the image geometry is possible before further calculations. This may be helpful, because rectification of the whole image is not efficient in the current context and may exhibit interpolation effects that impede segmentation. For image regions that correspond to pharmaceutical pills, rectification is feasible, however, because the amount of input data is much smaller.

For the computation (estimation) of the 3x3 Matrix $\mathbf{H_{IW}}$ at least $n = 4$ point correspondences have to be found between image plane ($\mathbf{p_{1I}}, \mathbf{p_{2I}}, \mathbf{p_{3I}}, \mathbf{p_{4I}}$) and the corresponding world plane ($\mathbf{p_{1W}}, \mathbf{p_{2W}}, \mathbf{p_{3W}}, \mathbf{p_{4W}}$) (see Figure 35 for the geometric alignment). In consideration of a possibility to realize such a rectification procedure efficiently in StbES, the minimum number of 4 points was chosen. Under the assumption of a rectangular alignment of target points, correspondences can be established by a heuristic, since the image geometry can be assumed to be relatively constant. Assuming the geometric alignment of Figure 28, the following procedure can be carried out:

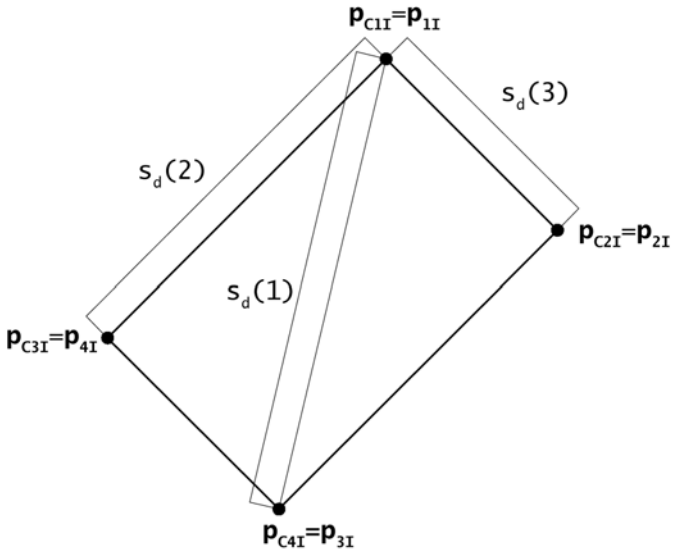

Figure 28: Exemplary geometric alignment for the determination of point correspondences (image plane)

- From four candidates in the image $\mathbf{p_{C1I}}, \mathbf{p_{C2I}}, \mathbf{p_{C3I}}$ and $\mathbf{p_{C4I}}$, find the top-left point and take it as $\mathbf{p_{1I}}$.
- Determine a descending sequence s_d of distances from $\mathbf{p_{1I}}$ to the remaining three candidates.
- Take the corresponding points starting from the maximum distance $s_d(1)$ as $\mathbf{p_{3I}}, \mathbf{p_{4I}}$ and $\mathbf{p_{2I}}$.

4.3.2 Measurement of Length and Width

The definition of length and width used within this work makes it necessary to determine the direction of maximum variance within a region where a pill is supposed to be. This corresponds to ellipse fitting, where the intersections of minor and major axis with the contour must be be computed in an additional step. Since distortions and differences in distance would deteriorate the result, all calculations have to be carried out on rectified points. Despite considerable saving in computation time, fitting cannot be carried out solely on the contour, because such a small amount of points would harm robustness.

By computation of the principal component analysis of the rectified coordinates within a region, the needed direction may be obtained as the eigenvector $\mathbf{v_1} = [v_1; v_2]$ that corresponds to the largest eigenvalue λ_1. The latter equates to the length (one side) of the major axis then (see [45]). As the current problem is two-dimensional, direct computation of $\mathbf{v_1}$ is possible from the entries σ_{ij} of the covariance matrix C (see [67]).

$$\lambda_{1,2} = \frac{1}{2}(\sigma_{11} + \sigma_{22} \pm \sqrt{(\sigma_{11} - \sigma_{22})^2 + 4\sigma_{12}^2}) \tag{45}$$

$$v_1 = \frac{\sigma_{12}}{\sqrt{(\lambda_1 - \sigma_{11})^2 + \sigma_{12}^2}} \tag{46}$$

$$v_2 = \frac{\lambda_1 - \sigma_{11}}{\sqrt{(\lambda_1 - \sigma_{11})^2 + \sigma_{12}^2}} \tag{47}$$

The length of the minor axis and the direction along it are then given as λ_2 and $\mathbf{v_2} = [-v_2; v_1]$. In order to obtain an estimate for the length and width of an object, it is necessary to project the boundary that makes up an object using the obtained vectors and $\mathbf{v_2}$. In fact, it is sufficient to project the world points that correspond to the convex hull, because they bound the extension of the object. If the vector $\mathbf{m} = [m_x; m_y]$ corresponds to the centroid of all world plane points that make up a region, a projection may be computed as:

$$\begin{bmatrix} xi_{W(p)} \\ yi_{W(p)} \end{bmatrix} = \begin{bmatrix} v_1 & v_2 \\ -v_2 & v_1 \end{bmatrix} \begin{bmatrix} xi_W - m_x \\ yi_W - m_y \end{bmatrix}, \tag{48}$$

where $[xi_W; yi_W]$ denotes a point in the world plane that belongs to the current region, and $[xi_{W(p)}; yi_{W(p)}]$ denotes its projected counterpart. An estimation of length and width may then be computed by computation of differences between minimum and maximum values of these coordinates.

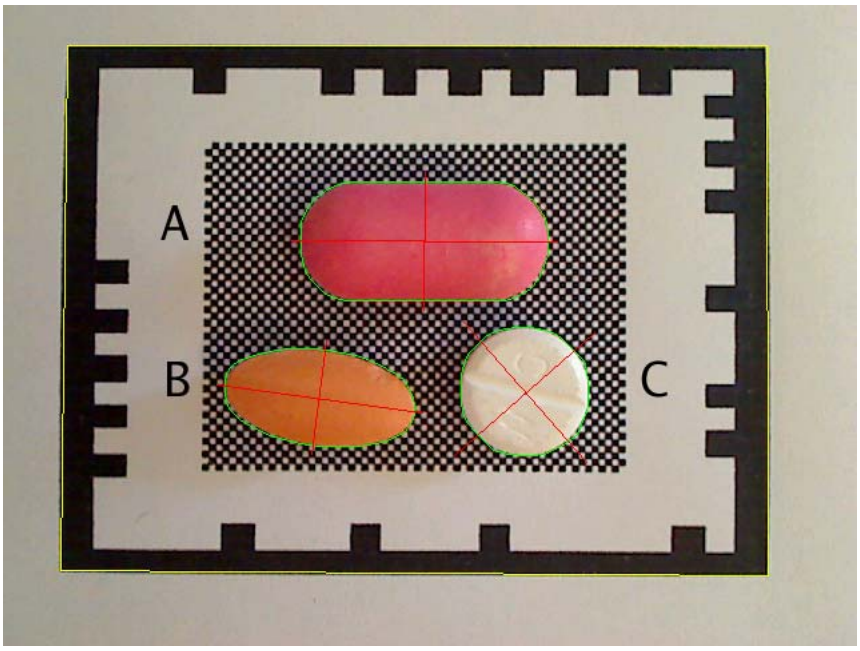

Figure 29: Visualization of measurement directions for three different shapes: A...oblong, B...oval, C...circular

The obtained estimates of various methods on data obtained by segmentation with a square size of $sq_w = 0.6mm$, are given in Table 5. In Figure 29 the visual result is shown.

Shape	**Method**	**Length [mm]**	**Width [mm]**
A (oblong)	nonius	21.35	10.30
	ell. fit	24.18	11.87
	proj. fit	22.00	10.33
B (oval)	nonius	16.20	8.00
	ell. fit	17.78	8.63
	proj. fit	16.96	8.20
C (circular)	nonius	11.08	11.08
	ell. fit	11.77	11.77
	proj. fit	11.40	11.40

Table 5: Size estimation results: shapes from Figure 29 ($sq_w = 0.6mm$, averaged values for circular shapes)

4.3.3 Considerations on Accuracy

The proposed method for the estimation of length and width shows differences from the ground truth, which was obtained using the nonius from the rapid

identification kit. In the following a number of reasons are discussed.
Radial distortions in image acquisition devices may have a considerable influence on accuracy. To account for this effect, a correction was implemented by warping the input image, and alternatively, by an appropriate coordinate transformation. To exclude additional warping, these tests were not carried out within StbES. Interestingly, the first method deteriorates accuracy. This may be attributed to the chessboard pattern on the target, that is sensitive to aliasing effects. A coordinate transformation did not considerably improve results either. The reason could be the autofocus mechanism of the camera, that may influence lens characteristics. So, the coefficients obtained from camera calibration cannot be used. On the other hand, objects may be assumed to have little offset from the center of distortion and do not extend over the whole image, so this influence will be limited in many cases.
The major source of error is the square size of the chessboard pattern, that limits the accuracy of segmentation. Under the assumption of a target that entirely fills the viewable area and does not exhibit other kinds of distortion, the expected error is dependent on the square size sq_w. As an upper bound, the error may be assumed to be double the square size then. The minimum usable square size sq_w is dependent on the image resolution. Tests have shown that about 3 pixels are necessary that correspond to one square length in the image. If we assume that pills may be correctly segmented under the previous conditions, the maximum error e_{max} can be estimated as follows:

$$e_{max} = 2 \cdot sq_w \tag{49}$$

With a square size of $0.6mm$, the expected error e_{max} is $1.2mm$. With images of 640 x 480 pixels, the square size may be reduced to $0.5mm$, giving an estimated error e_{max} of about $1mm$.
Accuracy is also dependent on the estimated homography H. Measurements should be made near the reference points to give more accurate results. With respect to an efficient StbES implementation, which makes use of the built-in tracker for target detection, the number of reference points and their positions cannot be altered, however.

According to these calculations, the accuracy required for the rapid identification kit cannot be reached with a resolution of 640 x 480 pixels.
Using an ordinary ruler as measurement tool, it is rather difficult to measure lengths below 1 mm, and the shapes of pills further aggravate measurements. For these reasons, the accuracy of the proposed method can be expected to be better than what can be obtained with a ruler. This assumption is supported by evaluation on a representative test set (see Section 6.4.2).

4.4 Color Estimation

In general, pharmaceutical pills may have arbitrary colors. In the rapid identification scheme a wide range of colors is *defined* by the corresponding names and a supplementary color chart (F1-F18) is necessary for distinguishing very similar tones. On-line databases also require the user to specify colors by their names. So, human perception and judgment of color is an essential part of this process. Consequently, a color estimation procedure should correspond to

human perception as good as possible. It is necessary to find an estimation procedure that may operate under different viewing conditions and thus recreate the phenomenon of chromatic adaptation (see Section 2.6.2) up to a certain degree. This means that methods for color correction and filtering of input data are needed. Again, the application of a target is beneficial in this case, because reference measurements can be made. All these operations need to be performed efficiently, in order to be computable on mobile devices within a reasonable amount of time.

4.4.1 Color Classes

Research in literature and on-line databases reveals that most pills have one or two colors, out of a set of 16 basic color tones (black, white and gray are treated like a color). Often, little saturated colors are used and multi-color pills are mostly divided along the minor axis. Some sources also list *clear* (i.e. transparent) as an additional color, which is not considered in this work, however.
An approach often applied with such problems is the definition of color classes by adaptation of point clouds in the color space used, which have been obtained from example data. This is not possible in the current case, due to the very limited amount of available samples. So, several example tones for each color class were collected using the available examples, images from on-line databases and literature. They were defined as sRGB triplets, because a device may be expected to deliver sRGB data, if no additional information is given. It must be noted that several classes are very similar and it is difficult to differentiate between them, even for the human observer. The mean color of each individual class is shown in Figure 30.

Figure 30: Mean colors for pill color classes (from left to right, starting from top): black, white, blue, beige, brown, gray, green, ocher, pink, violet, orange, peach, rose, red, cyan, yellow

These sequences of representative triplets for each color class form the basis for the creation of a color classifier.

4.4.2 Color Classification

As reasoned above, the required color classifier in the system to be developed must be efficient and deliver results consistent with human perception. The obvious approach for the estimation of color would require conversion into an approximately uniform color space such as CIE LAB. Then, the most similar

class needs to be determined by evaluation of an appropriate color distance metric. Unfortunately, such non-linear transformations and distance computations cannot be carried out efficiently on mobile devices yet. Alternatively it would be possible to apply a segmentation-based approach, such as region growing with a suitable similarity criterion. Application of a Mean Shift procedure seems less desirable, because this would have to take place in a uniform color space in order to fulfill the requirement of correspondence to human perception of color.

A more efficient approach is possible by using a precalculated lookup table (LUT) in the input color space (sRGB) for per pixel classification. Then, color classification for an arbitrary pixel may be carried out in constant time. A color classification result for the entire region may then be obtained by analysis of the corresponding histogram $h_c(i)$ of class labels, for example.

With lookup tables, savings in classification time come at the cost of additional storage and memory requirements. For practical reasons, the LUT size should be comparatively small. For the look-up tables within this work, the value s_{LUT} denotes the amount of entries for each channel in a 3D LUT. So, the amount of elements is s_{LUT}^3. Creation of a full sRGB look-up table for 24 bit images would result in a look-up table size of more than 16 MB, which cannot be handled on mobile devices in an efficient way yet.
In order to avoid costly conversions, the LUT should be defined for the input color space. Due to the small amount of samples and the fact that the (s)RGB space is not well suited to regular volumes, a different method for the computation of color volumes in the LUT is necessary. The entries for a rather small LUT can be computed by application of the ΔE_{CIE00} color distance metric in CIE LAB space (see Section 2.6.1). Then, a classifier that complies with human perception can be obtained. The sequence of operations for this task is as follows:

- Definition of representative sample colors c_{ij} per class. They need to be visually similar and should show a smooth course of color within a class.
- Assignment of a label l_i to the examples of class i.
- Definition of the desired LUT size s_{LUT} (equal for all dimensions).
- Conversion of all sample colors into CIE LAB space using the D65 white point (daylight).
- Iteration through all sRGB LUT entries, computation of the corresponding sRGB color in CIE LAB space and the distances d_{nn1} and d_{nn2} to the nearest samples by using the ΔE_{CIE00} color distance metric.
- Assignment of a label l to the current LUT entry after analysis of the distances d_{nn1} and d_{nn2}.

By populating the LUT in this manner, it is necessary to define two distances (thresholds), which may be used to mark entries that correspond to cases, where a decision is not possible or no suitable color may be found. In the first case, the entry is marked having *ambiguous* color. In the latter case, the entry is marked having *unknown* color. The conditions are as follows:

$$|d_{nn1} - d_{nn2}| < d_{amb} \tag{50}$$

$$d_{nn1} > d_{unk}, \tag{51}$$

where d_{amb} and d_{unk} are suitable thresholds.The proposed approach relies on the assumption that the colors which may be traversed by exhibiting the step size that is defined by s_{LUT} on each channel, show no or negligible perceptual difference.

As a first hint concerning the applicability of the LUT, the distribution of labels within it may be analyzed. In order to reduce memory requirements, we strive for the smallest LUT, however. If color classes are defined according to visual similarity and one of them is not represented in the LUT, the size is obviously too small. It may be assumed that an approximately constant relative amount of entries per class, starting at size s_{LUTC}, corresponds to a suitable LUT from this size on. In order to check, whether the distribution in the LUT is able to separate the color classes adequately, the sRGB colors that correspond to a specific color class may be visualized in a 3D plot. In an iterative procedure, it is possible to verify the distribution of classes. Analysis of the 3D plot and extension of the initial samples makes it possible to create an improved LUT then.

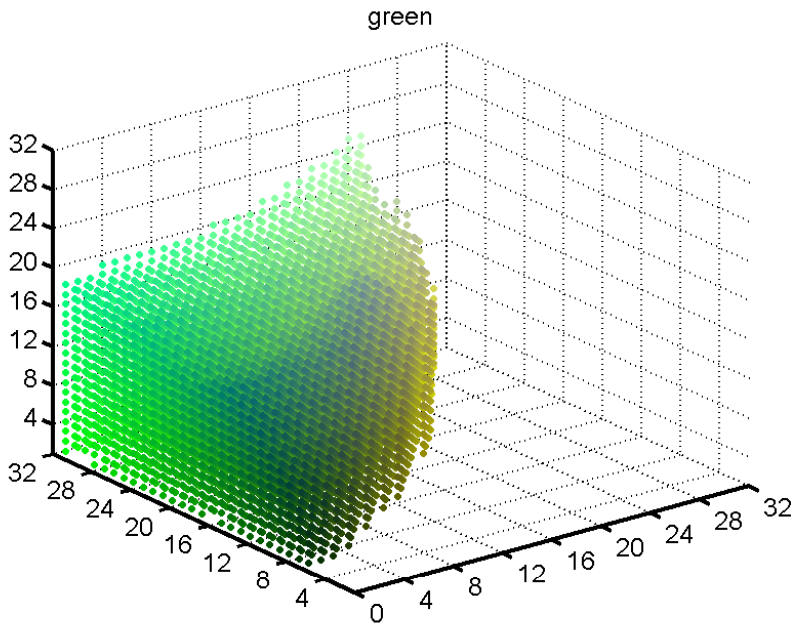

Figure 31: Exemplary 3D plot of class *green* as defined in a sRGB LUT with 32 entries per channel and 16 classes

In Figure 31 a plot for the color class *green* in a 3D sRGB look-up table with 32 entries per channel is shown ($s_{LUT} = 32$). This LUT structure divides the sRGB cube with a step size of about 8 units per channel and corresponds to

a LUT size of 32 KB (without administrative data). Close examination of the plots for the remaining classes showed that the approach succeeds in generating a suitable per pixel classifier. The result of such a per pixel classifier on a region representing a pill may be visualized by assigning the corresponding mean colors to the label image obtained by the classifier (see Figure 32 A and B).

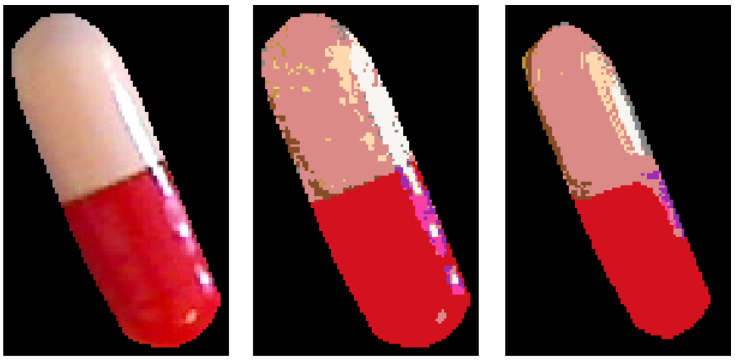

Figure 32: Per pixel classification: A...input region; B...result on non-filtered data; C...result on color smoothed data ($w_h = 6$)

In order to suppress interference with the chessboard pattern, a small part on the boundary of the original mask was discarded.

The required classification of up to two colors may then be obtained by analysis of the class histogram h_c. It is based on analysis of the relative amount of covered pixels in the region as well as the significance of the result, which is based on the distance between the first two entries of a sorted class histogram h_{cd} (descending order). The following metrics are used to decide on the color(s) of a region:

- coverage for a result with one color (label i):

$$c_i = \frac{h_{cd}(0)}{\sum h_c} \tag{52}$$

- coverage for a result with two colors (labels i and j):

$$c_{i,j} = \frac{(h_{cd}(0) + h_{cd}(1))}{\sum h_c} \tag{53}$$

- significance for a result with one color (label i):

$$s_i = \frac{h_{cd}(0) - h_{cd}(1)}{h_{cd}(0)} \tag{54}$$

- significance for a result with two colors (labels i,j):

$$s_{i,j} = \frac{1}{s_i} \tag{55}$$

A decision is possible, if one of these coverages is above a threshold c_{th}, because this assures a more meaningful result. Then, the products $c_i \cdot s_i$ and $c_{i,j} \cdot s_{i,j}$ are used to decide, whether one or more colors are present. If the first product is greater than the second, the result is the corresponding label i of the maximum entry in h_{cd}. Otherwise two colors are assumed and the labels i, j that correspond to the first two entries in h_{cd} are reported.
In Figure 33 a corresponding classification result is shown.

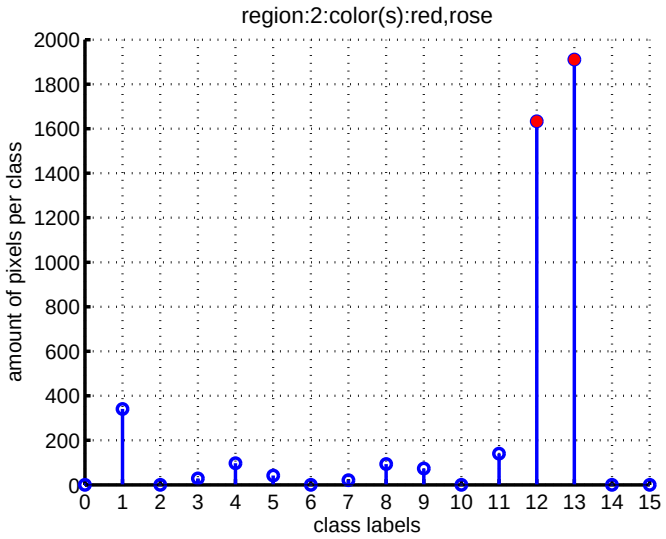

Figure 33: Histogram with classification result of the region from Figure 32 B

By visual inspection it is evident that the result (red, rose) corresponds to the colors of the pill. A decision may be particularly difficult in case of inhomogeneous lighting, however.

Some domain research reveals that in most cases only capsules have more than one color. As they are of oblong shape, this information can be used to setup the color classifier to detect more than one color only on capsules and improve the results.

4.4.3 Enhancement of Input Data

Color noise, highlights and shadows, especially on capsules, can degrade the result of the proposed classifier because these effects lead to wrong pixel classification results that are often inhomogeneous within the region.
In order to mitigate the problem of color noise and highlights, an efficient

method for *smoothing* of color is necessary. In general there are solutions that operate on each channel separately, or such that treat each pixel value as a vector that is composed of several channels. Application of a rank order filter such as the median filter does not seem appropriate, because the correction of impulse noise is not aimed for, but rather an overall *smoothing* of color. Thus an approach with separate treatment of color channels is proposed, because it is not necessary to explicitly preserve edges in the image. The process consists of the following steps and may work only if a mask of the region is available:

- Computation of integral images f_{IIc} for each channel c on the region defined by the bounding rectangle.
- Computation of a mean image f_{mwhc} for each channel c using the corresponding integral image f_{IIc}, according to the window size $2 \cdot w_h + 1$.
- Erosion of the corresponding mask with a square structuring element of size $2 \cdot w_h + 1$ and selection of filtered values that correspond to the mask.

For this specific application, the amount of artifacts can be assumed small, if the window size is relatively large. An erosion of the initial mask is necessary, because the operation processes data from the outside of the region, which must not distort the result near the border. From Figure 32 B and C it is evident that the influence of noise and highlights is reduced and the result is smoother.

Varying viewing conditions (different from daylight) were obviously not considered in the creation of the color classifier. Since this type of *distortion* will often occur in a real word scenario, it has to be accounted for. One approach would be to use a LUT that is adapted to a specific viewing illuminant. Another is to implement a method to correct the input colors based on certain measurements in the image. The first approach would require additional space for storage and possibly system memory while the second one may require considerable computational effort. Again, an efficient solution is necessary for mobile devices.

4.4.4 Color Correction

The created color classifier was designed to operate under viewing conditions that correspond to daylight (D65 white point). The non fixed setup as well as the fact that we strive for application on a mobile device, call for a practical solution. The result of the previously introduced color classifier with pixel data obtained under differing conditions is shown in Figure 34.

The pill is classified gray (see Figure 34 B), which does not resemble the color of the original object (a darker shade of green; see Figure 34 A). So, sufficiently different viewing conditions may lead to complete failure of the classifier. These include changes in brightness as well as the color lighting.
As with photography, a correction of the white point may give better results. Sometimes, not only white balance is performed, but a color calibration step. By analysis of colored areas on a reference card (color checker) a correction is possible, which is applied to the entire image then. In a mobile application, where the distance for image acquisition is very small, conditions may vary considerably within the image, however.

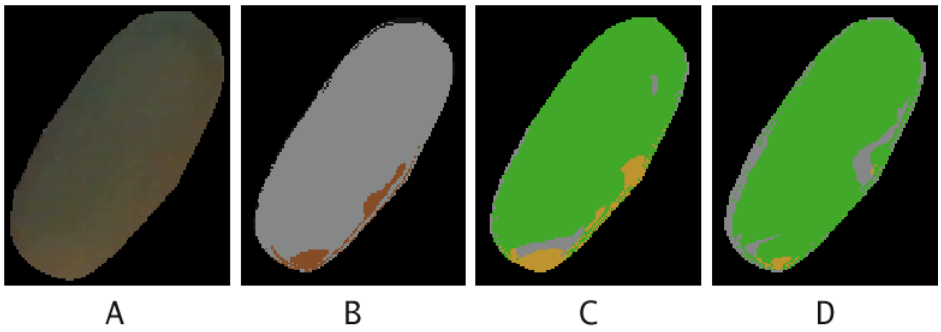

Figure 34: Color classification results with inappropriate lighting: A...extracted region; B...no white point correction; C...white point correction (Photoshop); D...white point correction (proposed method)

The need for a target in the current context demands a solution that is easily reproducible on a standard office printer and shows little aging effects, which is why the target was decided not to contain color at all. This makes white balance the method of choice. A local method is desirable because it may account for local changes in brightness and may allow a more efficient implementation.

In white balancing, several methods may be applied. Due to the requirement of computational efficiency, a linear method was chosen (see Section 2.6.2). Closer examination reveals that this makes it necessary to compute several costly matrix transformations, because correction ought to be carried out in *cone space* and the transformations usually involve floating point calculations.
With the proposed per pixel classifier, it is necessary to compute a correction for each pixel in the region. Despite the predicted limited accuracy, a correction in the input space by scaling the individual channels is proposed. To avoid distortions, the measured white point may not be too different from D65 lighting then. As a desirable side effect this approach implicitly addresses brightness correction and may be efficiently computed. Based on the measured white point $\mathbf{wp_{new}} = [r_{wpn}, g_{wpn}, b_{wpn}]'$, a correction of color values $[r_c, g_c, b_c]'$ (using white point $\mathbf{wp_{old}} = [r_{wpo}, g_{wpo}, b_{wpo}]'$) may be achieved by scaling of input RGB data $[r_{in}, g_{in}, b_{in}]'$. This may be formulated as a matrix multiplication that can be carried out using integer arithmetics.

$$\begin{bmatrix} r_c \\ g_c \\ b_c \end{bmatrix} = \begin{bmatrix} r_{wpn}/r_{wpo} & 0 & 0 \\ 0 & g_{wpn}/g_{wpo} & 0 \\ 0 & 0 & b_{wpn}/b_{wpo} \end{bmatrix} \begin{bmatrix} r_{in} \\ g_{in} \\ b_{in} \end{bmatrix} \tag{56}$$

As shown in the previous equation, an estimate of the current white point wp_{new} is necessary. Analysis of acquired images shows that estimation of a global white point is not sufficient, because non-uniform lighting is rather common for this type of setup. So, a local estimate is necessary.

In general it would be desirable to obtain a local white point estimate in the neighborhood of the pixel which must be corrected. Due to their small size, it is not practical to probe the white areas within the chessboard pattern in a robust manner, however. Since the target is assumed to be printed on white paper, the

area around the chessboard region may be used to get reference measurements of the white point. The values within the chessboard region may be approximated by an interpolation procedure instead. This area may be defined by four initial world points $\mathbf{p_{1W}}, \mathbf{p_{2W}}, \mathbf{p_{3W}}$ and $\mathbf{p_{4W}}$, which mark the corners of a rectangle. Their connecting lines shall be referred to as *correction lines* from now on. The necessary steps to obtain a white point estimate $\mathbf{wp_{new}}(\mathbf{x}, \mathbf{y})$ at location (x, y) within the chessboard region may me divided into two major steps.

- **Preparation:** Estimation of white points along the correction lines by transformation of their point coordinates $\mathbf{p_{ijW}}$ into image space, where i determines a correction line and j a point along it (once per image).
- **Estimation** at image position $(x, y)_I$: The white point estimate $\mathbf{wp_{new}}(\mathbf{x}, \mathbf{y})$ may be calculated by transformation of the coordinates $(x, y)_I$ into the world plane and projection of this location onto the connecting lines between the four initial points. This gives two distances d_{1W} and d_{2W} in the world plane, which may be used to scale the corresponding initial white point estimates $\mathbf{wp_{1Wr}}, \mathbf{wp_{2Wr}}, \mathbf{wp_{3Wr}}$ and $\mathbf{wp_{4Wr}}$ on their correction lines.

With the assumption of white target paper and negligible local noise, the points $\mathbf{p_{ijW}}$ taken at suitable distances, need to be transformed into image space in the first step and indices of the pixel values need to be stored. So, each point along a correction line in world space is assigned a white point estimate in the first step. Due to the rectangular shape of the defined area, projection is just reading the coordinates (possibly after a translation into the point of origin) and only one horizontal d_{1Wr} and one vertical distance d_{2Wr} is necessary for further computation. These distances define the white point values $\mathbf{wp_{1Wr}}, \mathbf{wp_{2Wr}}, \mathbf{wp_{3Wr}}$ and $\mathbf{wp_{4Wr}}$ on the four correction lines. Using these values and the two distances d_{1Wr} and d_{2Wr}, an estimate for location $(x, y)_I$ may be calculated by equally weighted linear interpolation in two dimensions (see Figure 35).

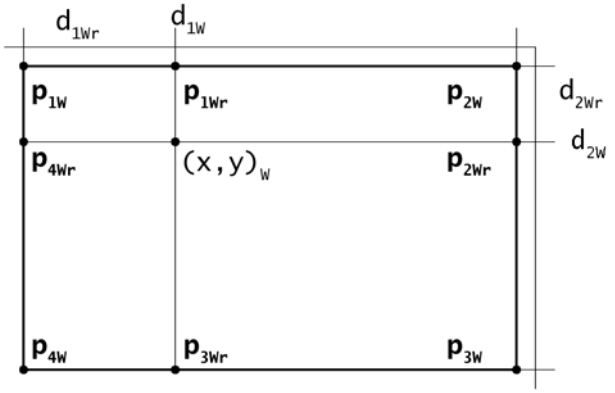

Figure 35: Illustration of geometry for white point correction

The value of the white point estimate $\mathbf{wp_{new}}(\mathbf{x}, \mathbf{y})$ for channel c may be calculated in the following way:

$$wp_{new}(x,y)[c] = 0.5(wp_{1W}[c]\frac{d_{2W}}{d_{2Wr}} + wp_{2W}[c]\frac{d_{1Wr}}{d_{1W}} + wp_{3W}[c]\frac{d_{2Wr}}{d_{2W}} + wp_{4W}[c]\frac{d_{1W}}{d_{1Wr}}) \tag{57}$$

If the initial preparation is neglected, the procedure roughly requires only one matrix multiplication on higher precision data (conversion to world space) while the other operations may be carried out on integers in general.

For the initial example image, there is little difference between the result of the proposed method (see Figure 34 D) and the result on input data that was corrected by a popular image processing software (see Figure 34 C).
Despite these results, it is important to note that the approach cannot be applied if large deviations from the LUT white point have to be corrected. The amount of necessary scaling can be recorded during the process and it seems reasonable to inform the user about an inappropriate input image then.

5 Mobile Phone Prototype

5.1 Preface

A prototypical system was implemented using the algorithms described in the previous chapters as well as existing functionality from the StbES framework (see [52]). This allows to demonstrate the applicability of the proposed methods for segmentation and feature estimation on mobile devices. The StbES environment has the advantage that it is highly optimized and already includes some of the algorithms which are required by the proposed method. The communication capabilities of StbES make it possible to realize a prototype that is able to query an on-line database and present the query results in a convenient way (see Figure 36).

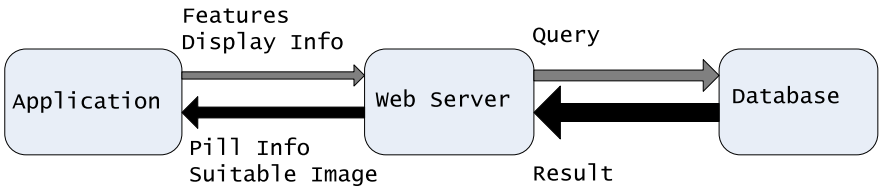

Figure 36: Diagram of data flow within the system

From an acquired camera image, object properties are estimated using the algorithms described previously. These may be used to query a database from within the application. For each candidate, the database will answer with textual information and an image for visual verification. This has to be carried out efficiently, minimizing the effort for communication and preparation of data.
An additional piece of middleware was developed that runs on an ordinary web server. Besides minimizing the amount of transferred data, it serves as an interface for arbitrary databases.
The prototype demonstrates the application of computer computer vision aided identification of pills based on the estimated features and allows its evaluation on real online databases.

5.2 StbES Frontend

In the StbES frontend a pill recognition session is run consisting of a sequence of steps where each one depends on its predecessor (as illustrated in Figure 37). To a large extent these steps correspond to the structure of the actual implementation.

All computer vision related processing is performed on the client, whereas pill information is retrieved in suitable form from an external source.

Necessary parameters for the pill recognition application are read from an additional configuration file at startup. Image source, location of the server, maximum amount of results, as well as the mode of operation may be configured.

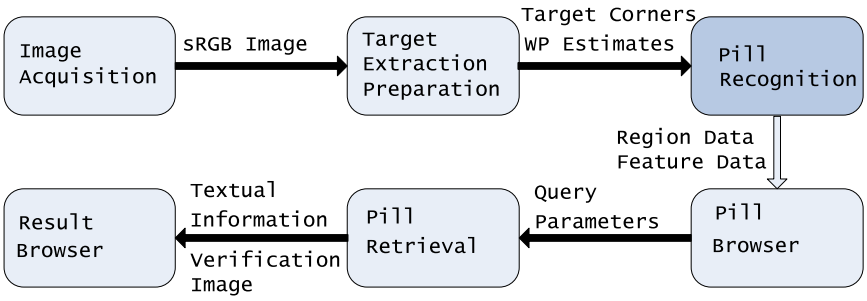

Figure 37: Course of the pill recognition procedure

The entire application can be operated using five buttons in order to simplify a mapping on the directional stick found on most mobile devices. This operational concept has been realized using a state pattern [24], since it provides a level of abstraction and future modifications may be carried out with reduced effort then. In the following, the relevant steps are described in more detail.

5.2.1 Image Acquisition and Target Extraction

Suitable images for the application may be obtained from an autofocus camera, since fixed focus cameras are unable to give a sharp image of suitable size. The application at hand may process a 24 bit sRGB image obtained from video input within StbES or from a file.

For reasons of practicability, it seems justified to choose the outer dimensions of the reference rectangle similar to the size of items that may be stored in a wallet. This allows to store the card at the back of a mobile device, if desired. In view of this possibility, the dimensions of a credit card (85 mm x 55 mm) were chosen. Bending the card has to be avoided, since that violates the assumptions of a homography. For an aspect ratio that corresponds to that of the capture resolution, the remaining dimension of the target area may be computed as follows:

$$x = \frac{4}{3} \cdot 48mm = 64mm$$

In this calculation the height y is set to 48mm because the added surrounding white area makes the detection of the marker more robust. The usable area for the recognition of pills has a size of 38mm x 30mm then. See Figure 38 for an image of the card.

For gaining robustness in the detection of reference points for size estimation, the built-in tracker of StbES is manually instantiated within the application. When the marker is visible, the four corners can be obtained from within the tracker object. With knowledge of the real dimensions it is possible to compute a homography using the mechanics of StbES. With this information, the coordinates for the outer white point estimates and their values are computed. In order to reduce the amount of input data for segmentation and feature estimation, a rectangular region of interest (ROI) is extracted.

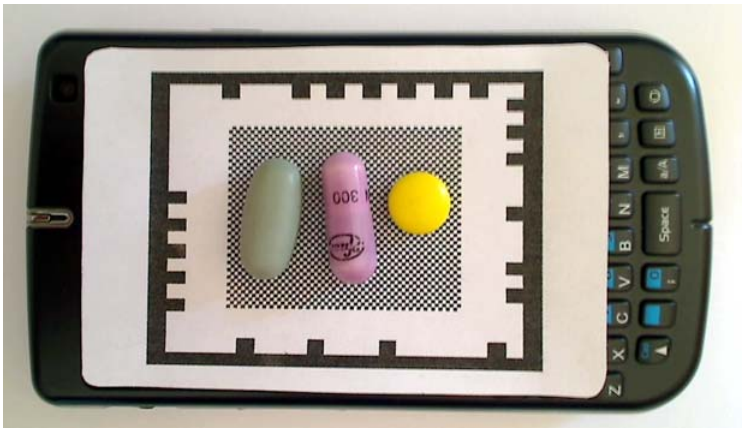

Figure 38: Card with frame marker positioned on smartphone

5.2.2 Pill Segmentation and Feature Extraction

In this step, all necessary processing for the extraction of regions as well as feature estimation is carried out. For the latter, suitable training data is necessary. The segmentation procedure proposed in Section 3.4 required the implementation of additional algorithms, since no suitable StbES counterparts for various functionality used in early prototyping were available. In addition, a method for incorporation of shape training data and the import of the color lookup table into the StbES application had to be found. As the final outcome of this step, a set of relevant regions is stored along with the estimated individual features.

Local adaptive thresholding on integral images and morphological operations, as required for segmentation, can be realized rather efficiently with little difficulties. The same holds for feature extractors and classifiers, as described in Section 4. Mainly in segmentation, the implementation of individual solutions was necessary, as listed in the following.

- Segmentation (region labeling, boundary extraction): An efficient linear-time method that combines region labeling and contour computation in a single procedure was implemented after [8].
- Segmentation (boundary enhancement): An algorithm for computation of the convex hull was implemented after [26], since it is simple but efficient and allows a concise formulation.
- Segmentation (mask computation, flood fill): Some research lead to the conclusion that a stack-based implementation is suitable in the current application, in particular for reasons of memory usage (see [63]). In order to save processing time, it is carried out on the individual contours.

Shape classifiers and color look-up tables need to be available in the StbES application at runtime. Training of these classifiers for arbitrary shapes and colors is carried out beforehand. Due to the fact that this data will not be changed during program execution and for reasons of simplicity, the data is included into the compilation process as hardcoded sources. Although the data could also be

loaded on startup, reading from external files would significantly increase the startup time of the application and is therefore avoided.

The set of characteristics calculated individually for each region contains size (length and width) as well as color and shape estimates based on the set of classifiers trained and loaded previously.

5.2.3 Pill Browser

For verification of the results from the previous stage, the user is provided with some graphical visualization and textual information for each individual pill. At a glance it is possible to verify segmentation, color estimation and the directions used in the measurement of size (see Figure 39).

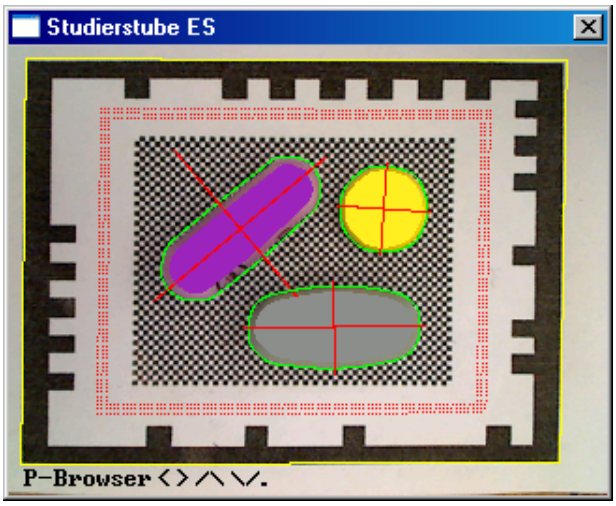

Figure 39: Visualization of feature estimation: boundaries, colors, directions of measurement

While the result of color classification is directly visible, hints on the quality of shape and size estimation are given by the course of the object boundary and the directions of measurement lines. Detailed information concerning the estimated features of the considered pharmaceutical pill is presented in the pill browser (see Figure 40). In case of an error the results may be quickly altered.

Using three buttons, all known classes and sizes may be chosen manually as query parameters as well. When color classes are changed, the visualization is updated with the corresponding mean colors of the new classes. After visual inspection a communication session can be initiated by the user for querying an online database about the selected pill. A closer description of this communication process is postponed until Section 5.3.

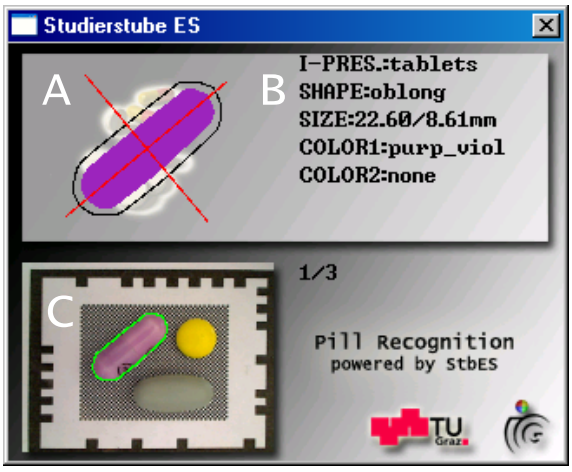

Figure 40: Pill browser: A...visualization of result, B...detailed information, C...display of active pill

5.2.4 Result Browser

The result browser serves as a means to provide the user with information about possible candidates that were obtained as a query result. In Figure 41 the exact match for the violet hard capsule in Figure 39 is selected.

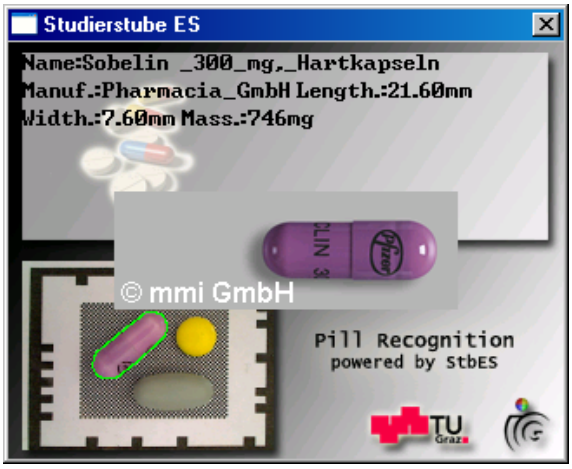

Figure 41: Result browser: textual information and example image for visual verification (violet capsule from Figure 39)

In the upper part of the screen the textual information retrieved from the database is shown. Usually, this textual information contains the name of the medical product, its manufacturer, and other characteristics, like its length, width and the mass of the candidate. If images are available in the online

database these can be retrieved on demand which reduces the initial round-trip time.

5.3 Database Connection

As shown in Figure 36 and as already mentioned above, the StbES application will start a communication session with a HTTP post request towards a web server running a suitable piece of software. For this prototype the necessary functionality was implemented as a script in object oriented PHP [25] (Hypertext Preprocessor).
The aforementioned script is responsible for communication with a suitable database which supports the estimated features. In general, an arbitrary database may be used for the retrieval of pill information. If additional information is provided by the database, such as the exact ingredients, these can be additionally forwarded to the application if desired.
After querying the database the results are stored in an XML [26] document which is transmitted to the client. It contains status information as well as textual information and image sources for all possible candidates. The script will prepare all images on the server in advance (resolution as well as orientation), so that no further processing is necessary on the client. The user may retrieve the corresponding image in the result browser, if desired. All communication with the (on-line) database remains transparent for the StbES application. As long as no additional pill information is desired, an arbitrary database may be used without changing the StbES frontend.

5.3.1 Exemplary Database Connection - The Identa Database

As an exemplary database connection, the Identa database[27] was chosen. Since the Identa database is intended to be queried by a human operator through a web form, all involved steps must be emulated by the server script. In this special case, a login was required over a secure connection, which, at the time of creation, could not be realized with the built-in communication facilities of StbES. The proposed architecture allows to realize all necessary functionality on the server.
Upon a suitable HTTP post request, the following tasks are carried out:

- Parameters are checked.
- Database login is carried out.
- A query is issued.
- The result is examined and all textual information is aggregated.
- All corresponding images are retrieved and prepared.

The login procedure required the incorporation of cURL[28] (Client for URLs), because a secure HTTP connection needs to be established for this purpose and

[25] `http://www.php.net/manual`
[26] Extensible Markup Language
[27] `http://www.gelbe-liste.de/pharmindex/identa`
[28] `http://curl.haxx.se`

```
<pillservice_result>
  <status> </status>
  <pillservice_subresult>
    <name> </name>
    <manufacturer> </manufacturer>
    <width>
      <number> </number>
      <unit> </unit>
    </width>
    <length>
      <number> </number>
      <unit> </unit>
    </length>
    <mass>
      <number> </number>
      <unit> </unit>
    </mass>
    <img_src> </img_src>
  <\pillservice_subresult>
</pillservice_result>
```

Figure 42: Structure of the result document transmitted to the StbES application (for Identa)

PHP provides no native support for that. If this step is successful, a query is issued towards the database. Depending on the result, additional communication may be necessary, since the answer is an ordinary HTML[29] document that may contain links pointing to additional results. Once they have been aggregated, all images are retrieved, prepared according to the given parameters and stored on the server. This means that no further communication is necessary for the current recognition session. The structure of the final XML document which is sent back to the client is shown in Figure 42.

For testing, the script was set up on a local Apache web server [30]. It may work on an arbitrary server as well, as long as support for PHP and cURL is there.

5.4 Application on a Smartphone

In the StbES framework an application may be compiled for several platforms. The flexible architecture of StbES allows to create a version for Windows Mobile based smartphones with manageable effort. With certain modifications to the original application it is possible to compute the entire pill recognition procedure in reasonable time on mobile devices.

[29] Hypertext Markup Language

[30] `http://projects.apache.org/projects/http_server.html`

5.4.1 Mobile Phone Specific Modifications and Optimizations

As discussed in Section 2.1.1, there are additional challenges when developing computer vision applications for mobile phones. Although the prototype did perform well on the development platform, the following modifications were necessary for an implementable system.

- Efficient Processing of Higher Precision Data: Although it may be possible to let the compiler translate all floating point calculations to integer arithmetics, there is usually a considerable deterioration in performance. A remedy is the application of fixed-point types, because they do not impact performance as dramatically and provide a level of abstraction. In the application at hand it is not possible to entirely do without such calculations, since the computation of a homography and the subsequent transformation of coordinates cannot be carried out on integer types.

- Preallocation of Memory: Tests on the mobile device revealed that subsequent allocation of memory severely degrades performance. The amount of slots for storing point coordinates in a vector structure are explicitly preallocated whenever possible.

- Image Acquisition: Images of 640 x 480 pixels cannot be captured using the built-in camera in video mode when working on specific smartphones. It is not intended to deactivate the video input within StbES either. Otherwise the camera application of Windows Mobile 6 could have been used for this purpose. This means that images of 320 x 240 pixels taken from live video are processed in such cases. By modifying the config file, it is possible to read and process images of higher resolution on these devices, however. As more recent smartphones can deliver video at 640 x 480 pixels, the application would not be limited in this regard.

- Visualization: The conventional display resolution in the application at hand is typically smaller than the intended capture resolution of 640 x 480 pixels. This means that the input image needs to be scaled for display on the mobile device. In addition, the coordinates of all points that need to be drawn on screen have to be scaled, too.

With these modifications and the corresponding changes in the configuration files, the application may run on a Windows Mobile powered smartphone (see Figure 43).

The retrieval of pill information is possible using any of the built-in communication facilities of the device, provided that the server part was properly configured for web access.

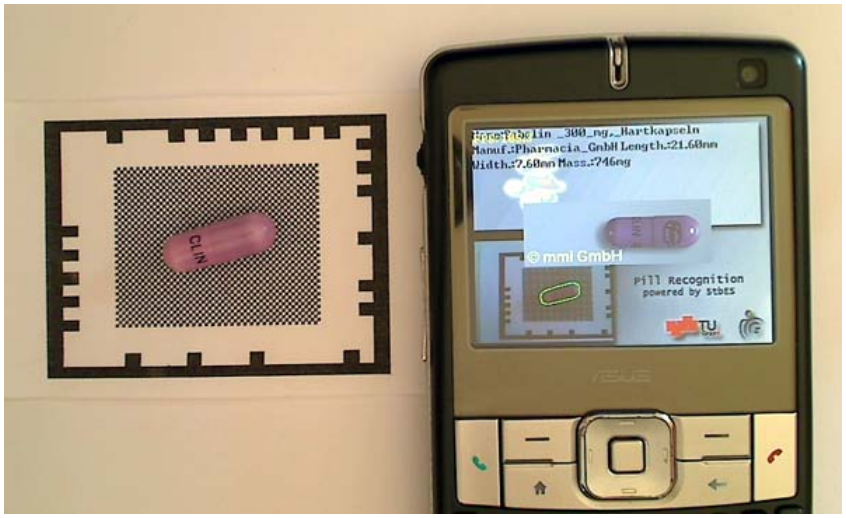

Figure 43: StbES pill recognition running on the Asus M530w smartphone (see Section 6.2.2)

6 Evaluation

6.1 Preface

Within this section an evaluation of the created pill recognition system is described and discussed. In the current case, evaluation serves the following main purposes:

- Determination of performance in shape and color estimation.
- Determination of deviations in size measurement.
- Determination of optimum system parameters.
- Determination of runtime for the whole system and individual parts.
- Estimation of system performance in a pill retrieval scenario.

To be able to demonstrate and evaluate the whole recognition procedure, the Identa on-line database was chosen to serve as the backbone for the application. This means that training of feature estimators needs to comply with Identa categories. First, each individual estimator is evaluated on suitable test data, while iterating through a set of parameters for the estimator, where possible. Then, system performance is evaluated with optimum system parameters. Retrieval performance on a reference database is investigated when using optimum system parameters. In this step, various combinations of features are evaluated. Finally, retrieval performance is evaluated on a subset of samples using the Identa[31] online database.

6.2 System Parameters

For the purpose of evaluation, the StbES application is called with known input images. For automatic processing of classification results in Matlab, XML output was added to the StbES application. In such a result file, information from the individual feature estimators is stored. Besides, additional information such as region sizes and runtime for individual steps are included. With the same image, different results may be obtained, depending on the current system parameters such as PGH size and LUT size. If not noted otherwise, all tests are carried out on images with a resolution of 640 x 480 pixels.

6.2.1 Training

The Identa database supports a subset of possible pill shapes as query parameters. The shape categories represent circular, oval, oblong and special shapes. The database also contains concave pills and additional forms such as suppositories, which are not considered in this work. In Figure 44, examples for each category are shown.

In order to create a suitable training set for the Identa database, the examples of the shape database (see Section 4.2.1) were categorized to be in accordance with the shape classes of Identa. The final training set is described in more

Figure 44: Examples for Identa shape classes: A...special class; B...oval class; C...oblong class; D...circular class

Class	Amount
circular	4
oblong	8
oval	10
special	16
sum	**38**

Table 6: Shape training data: Identa

detail in Table 6. Then, Pairwise Geometric Histogram features of suitable dimensionality were computed and stored for reference, as described in Section 4.2.2. Due to the specific properties of a PGH, the minimum amount of entries is 4. The maximum amount of entries for a square PGH within this evaluation is fixed at 256, because program size should be sufficiently small (runtime may be another reason). For the same reason, the maximum amount of entries per channel in the color LUT was set to 42. The next reasonable LUT size per channel would be 52, which is already impractical because the amount of data would be more than 137 KB.

In the Identa query from, 14 colors are differentiated by their names as well as a very small number of corresponding tones. There seem to be consistency problems within the classes, since several examples do not seem to fall into the corresponding color classes, as obtained by visual inspection. Due to the fact that the available samples do not cover the Identa database, no verification or correction is possible. This categorization should not be used in the creation of a color look up table, because it would lead to categories of questionable perceptive similarity. E.g. the classes *rose* and *peach* contain examples of very similar tones. In order to improve the significance of color estimation, such tones were merged into a single class (see Figure 45 for the corresponding mean colors).

[31] `http://www.gelbe-liste.de/pharmindex/identa`

Figure 45: Mean colors for Identa classes (from left to right, starting from top): black, white, blue, beige, brown, gray, green, ocher, pink/violet, orange, rose/peach, red, yellow

The mean colors of Figure 45 were computed from the generic categories of Section 4.4.1. These were merged were necessary to represent a perceptively similar categorization.

6.2.2 Evaluation Platforms

The main platform for evaluation is a dual-core laptop with 1.8 GHz and 2.5 GB of main memory, running Windows XP. For image acquisition a standard web-cam with autofocus is used (Logitech Quickcam Pro 9000 [32]).

Mobile performance is evaluated on an Asus M530w smartphone [33], running the Windows Mobile 6 operating system. From the point of the application at hand, the following specifications are of interest:

- Camera: 2 mega pixels with autofocus (video resolution: 320 x 240 pixels)
- CPU: Marvell PXA270 (416 MHz, no floating point unit)
- Memory: 64 MB RAM, 256 MB Flash
- Screen resolution: 320 x 240 pixels

Besides supporting several standards for mobile data communication, it features a USB connection as well as an integrated wireless networking device. Although it is clear that this is not a state-of-the-art device (introduced in 2007), it allows to get an impression of the expected performance on common devices.

6.3 Datasets

6.3.1 Reference Data

In the proposed evaluation procedure, reference data for the captured pills is necessary. For this purpose, each pharmaceutical pill was manually classified into the appropriate categories for shape and color. The length and width of each sample were measured using a nonius. All collected information about the samples was subsequently stored for later use (see Table 7 for a description of contents). Despite its small size, the set contains pharmaceutical pills with the most current shapes, colors and dimensions. In Table 8 a detailed listing of the distribution of colors for single-colored pills is given.

[32] http://www.logitech.com
[33] http://www.asus.com

Shapes	*circular*	*oval*	*oblong*	*special*
	41	26	33	8
Colors	*single*	*multi*		
	98	10		
Sizes [mm]	*min. length*	*min width*	*max length*	*max width*
	5.68	5.68	18.07	18.07

Table 7: Description of the global reference set: shape, colors, size

black	*white*	*blue*	*beige*	*brown*	*gray*	*green*
1	29	6	11	4	1	5
ocher	*pink/violet*	*orange*	*rose/peach*	*red*	*yellow*	-
6	3	6	11	5	10	-

Table 8: Description of the global reference set: single-colored pills

6.3.2 Test Data

Test images for each example in the global reference set were captured at differing lighting conditions. In a realistic scenario, the environment may be expected to contain light sources, that are similar to daylight or fluorescent light. The latter can be assumed to be the common light source in medical environments. All pills were captured at a resolution of 640 x 480 pixels, using the card of Section 5.2.1, which has a chessboard length of 0.6mm.
The filename of each test image was stored at its corresponding entry in the global reference database. This means that the example is labeled, which makes it possible to carry out the proposed evaluation procedure. Depending on the aspect that should be investigated, suitable reference sets were created. In Table 9 a description of their contents is given. This means that one representative

Name	**Instances**	**Lighting**
Set D	108	daylight
Set F	108	fluorescent
Set DF	216	both

Table 9: Description of the evaluated reference sets: D...daylight, F...fluorescent lighting, DF...both

image for each sample in the global reference set is included for set D (daylight) and set F (fluorescent lighting). For experiments, where the lighting conditions are secondary, both sets are used.

6.4 Results

6.4.1 Shape Estimation

Shape estimation performance is evaluated with square Pairwise Geometric Histograms with a total amount of entries between 4 and 256. With shape estimation, it seems more realistic to determine recognition performance with various types of lighting within one test set, since the implemented segmentation method can be assumed to be sufficiently invariant. So, the corresponding images of reference *set DF* were processed in the StbES pill recognition application for this task. The input resolution may have a larger impact on shape estimation, since the accuracy of segmentation is directly related to it.

Suitable PGH sizes may be determined by investigating the dependency of the recognition rate on the corresponding PGH size. In Figure 46 the characteristics of the recognition rate (sensitivity) is shown, when processing *set DF*. Figure 47 shows the corresponding ROC plots. In the determination of values for the ROC plot, examples which could not be processed in the application, are not considered. They have the expected influence on recognition rates, however. A detailed listing of recognition rates per shape class, obtained with a PGH configuration of 12 distance and 12 angle bins, is shown in Table 10.

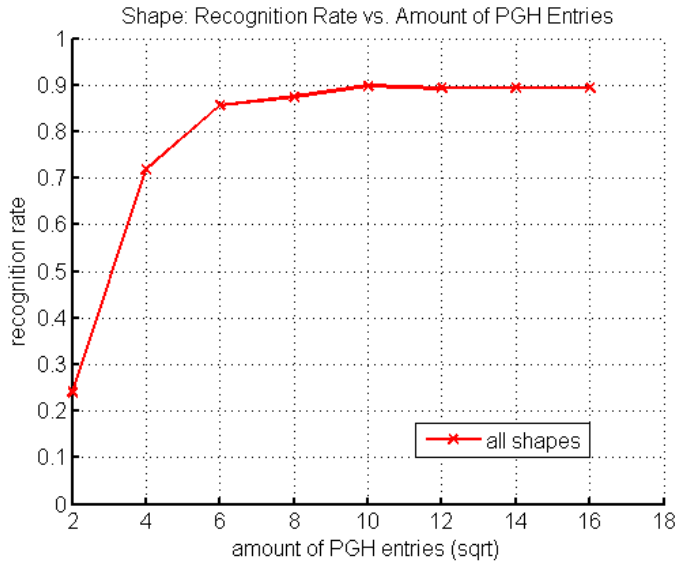

Figure 46: Results for shape estimation (640 x 480 pixels, all shapes): recognition rates with various PGH sizes

Problems with certain oval and oblong shapes are evident in the results. Several examples have similar shape, and a correct classification is often difficult, even

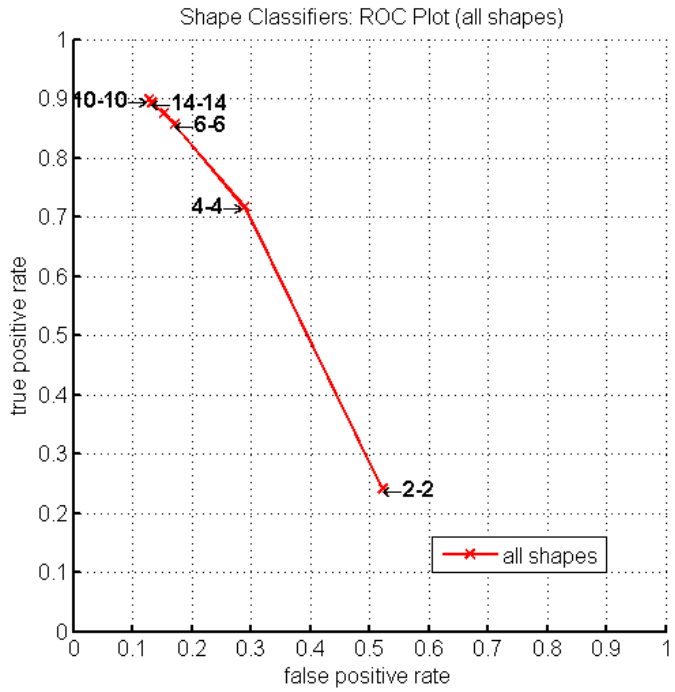

Figure 47: Results for shape estimation (640 x 480 pixels, all shapes): ROC plots from various PGH sizes

Resolution	Class	Recognition Rate
640 x 480	circular	0.9634
	oval	0.7692
	oblong	0.9384
	special	0.7500
	overall	**0.8930**

Table 10: Individual shape recognition rates on *set DF* (PGH D12-A12)

for the human observer. In this case, the course of angles along the border is often very similar. This means that artifacts of segmentation have a stronger influence on the result. Better results could be obtained by increasing the angular resolution of the PGH, provided that the boundary was obtained precisely enough.

From the gathered data it is evident that the best recognition rate is reached at a PGH size of 10 distance and 10 angle bins (true positive rate: 0.90). A robust classifier can be obtained with a PGH size of 12 distance and 12 angle bins (true positive rate: 0.89). A consideration of the corresponding false positive rate (0.13), qualifies this classifier for application in the pill recognition procedure. Although a smaller PGH corresponds to considerable savings in storage, the larger PGH size will increase robustness.

6.4.2 Size Estimation

The estimation of object length and width is directly dependent on the result of segmentation. As with shape estimation, accuracy is limited by the selected chessboard square length and the capture resolution.
In this experiment, deviations from the metric data in the reference set were recorded and the magnitudes of deviations across the set were analyzed. Mean value and standard deviations define a range for size information which can be used as an indicator for the quality of the measurements.
It is not practical to use the signed minimum and maximum deviations in such an application. The reason is that in the implemented segmentation method, deviations in the location of the object boundary may occur in arbitrary directions. In addition, there is no sequence of measurements for a single instance in this experiment. So, an approach with mean and standard deviation on magnitude values allows to specify queries more precisely on average.
The method employed for computing length and width (see Section 4.3.2) is dependent on object shape. So, a separate analysis of deviations per shape class is carried out.

In Table 11 the computed deviations for length and width measurement on *set DF* (daylight and fluorescent light) are shown. Obviously the range defined

Shape	**Descr.**	**Δ Length [mm]**	**Δ Width [mm]**
all	mean dev.	0.36	0.24
	std. dev.	0.30	0.19
	min dev.	0	0
	max dev.	1.38	1.37
circular	mean dev.	0.27	0.27
	std. dev.	0.25	0.24
	min dev.	0	0.01
	max dev.	1.38	1.37
oval	mean dev.	0.43	0.23
	std. dev.	0.32	0.17
	min dev.	0	0
	max dev.	1.21	0.72
oblong	mean dev.	0.43	0.21
	std. dev.	0.33	0.16
	min dev.	0	0
	max dev.	1.28	0.62
special	mean dev.	0.32	0.24
	std. dev.	0.31	0.14
	min dev.	0.03	0
	max dev.	1.24	0.46

Table 11: Deviations for size estimation on *set DF* at 640 x 480 pixels (computed on magnitude values)

by mean and standard deviation values agrees with the previously estimated maximum value of twice the chessboard square length (see Section 4.3.3. In fact, deviations lie within or slightly above one square length for all shapes. The maximum deviations for length and width across the whole set are slightly

higher, however (outliers).
The deviations differ slightly between shape classes. Circular shapes may be measured with the least deviations. The reason could be that errors in the directions of measurement have little influence in this case. In general, the magnitude of deviations in length is larger than those in width, with the exception of circular shapes. Besides, differences in the directions of measurement have a larger influence, when the point of measurement is farther away from the centroid.

From Table 11, query ranges for size measurement can be obtained, when processing images of 640 x 480 pixels. Based on the ranges defined by the mean and standard deviation values obtained for each shape class, the final value for length deviation is about $0.7mm$ and that for width deviation is $0.5mm$. With respect to the mean values for length and width in the global reference set, this corresponds to a mean deviation of 5.2% in length and 6.2% in width.

6.4.3 Color Estimation

For the determination of an optimal LUT size s_{LUT}, various amounts of entries per color channel were evaluated (see Table 12). The small overall amount of

s_{LUT}	Δch	Size [KB]	RR D	RR F
6	42.7	< 1	0.2041	0.1939
8	32.0	< 1	0.3265	0.2857
10	25.6	< 1	0.3469	0.3571
12	21.3	1.7	0.4388	0.4898
16	16.0	4.0	0.7449	0.7449
22	11.6	10.4	0.7857	0.7959
26	9.8	17.2	**0.8367**	**0.8367**
32	8.0	32.0	0.8367	0.8265
36	7.1	45.6	0.8367	0.8367
42	6.1	72.4	0.8163	0.83673

Table 12: Evaluated color lookup-tables (24 bit color data): recognition rates for single-colored pills (relative the total number of single colored pills), RR...recognition rate, D...daylight, F...fluorescent lighting

samples translates to an even smaller amount of samples per color class (see Table 8). Despite the fact that each color class is represented in the reference sets, an analysis of individual rates would not be meaningful. For the same reason, an individual analysis for multi color pills is would not give meaningful results (only relevant for oblong shapes). Under the assumption that the distribution of colors in the reference sets roughly resembles the real distribution for pharmaceutical pills, an evaluation is considered meaningful.
For reasons of usability, color classification in the application is only given, if enough evidence for a meaningful result is available (reduction of false positives). If no decision is possible, a class called *unknown* is assigned. Such examples are treated as having a wrong color classification result.
Under the assumption that the reference sets approximately represent the expected frequencies across pharmaceutical pills, an estimate of performance can be provided.

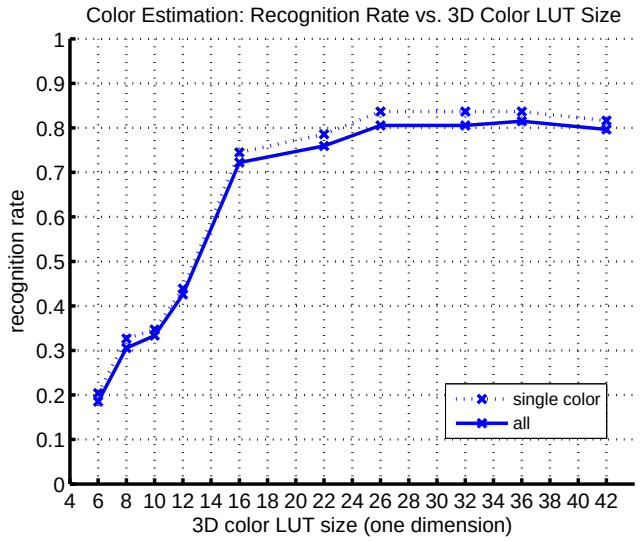

Figure 48: Results for single color estimation (640 x 480 pixels, all colors): set D (day light)

In Figures 48 and 49, the characteristics of recognition rates for day light and fluorescent light, when using the specified LUT sizes, is shown. Detailed values are presented in Table 12. Before giving an interpretation of these results, we have to keep in mind that some color classes or certain examples thereof are difficult to differ (beige, yellow, rose/peach). Inhomogeneous lighting within a region has a particularly negative influence when classifying such pills, and the recognition rate deteriorates.
The best recognition rate of 0.8367 is obtained with both light environments at $s_{LUT} = 26$. This rate is given with respect to all single colored pills in the reference sets. Results for multi color classification may not be analyzed separately, due to the small amount of available samples. The combined recognition rate (see Figures 48 and 49) and is lower than the one for single color pills, because it is given with respect to the entire reference sets.
The rate for identifying a pill as having more colors may serve as an indicator for the performance of two-colored pills, however. For a LUT size of $s_{LUT} = 36$, the multi color detection rate is ≥ 0.8 for both lighting environments. This LUT size is considered optimal with respect to the obtained results, because it may be seen as a trade-off between recognition rates, multi color detection rate and overfitting.
For day light, classification performance seems monotonous at first, but begins to degrade slightly with larger LUT sizes. With fluorescent lighting, the curve

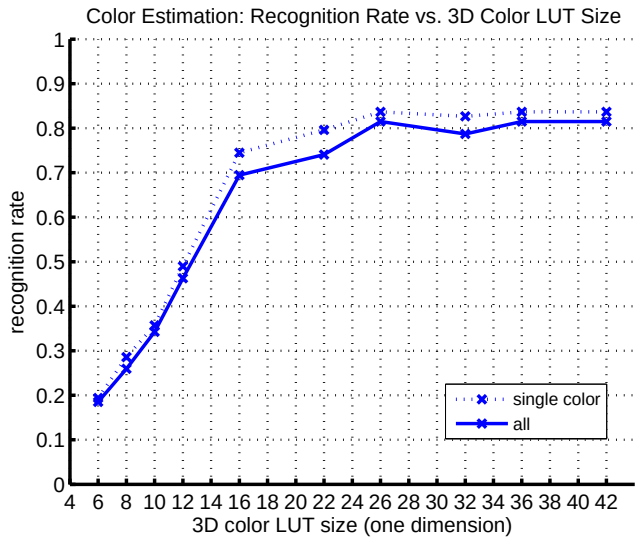

Figure 49: Results for single color estimation (640 x 480 pixels, all colors): set F (fluorescent light)

of recognition rates is not monotonic, after the best recognition rate has been reached. This effect may be caused by the employed white balancing method, where color distortions may occur in environments exhibiting considerably different lighting conditions than natural light. Another reason may be the method employed for LUT creation. Since the amount of samples was very small, the amount of example colors per class is also limited. As a meaningful color chart containing the tones for each color is not available, the representation might not be entirely smooth. A coarser LUT resolution has a smoothing effect. Thus, abnormalities are likely to occur with higher resolutions per channel, as is the case in this experiment.

6.4.4 Runtime

System runtime is an important aspect in the problem at hand. The pill recognition procedure must be sufficiently fast, because usability and acceptance of such a system would suffer otherwise. When neglecting the time needed to make manual corrections and to select the exact match, runtime is made up of two major parts. First, the pill recognition application will processes the acquired image and estimate features. The second part consists of querying the on-line database and presenting the results. While runtime of the first part can be determined using available test data, the second one is dependent on network latency, as well as the amount of results. Due to the fact that a lot of server side processing is necessary in case of Identa, which is not representative of a dedicated solution, runtime considerations for this part are not meaningful.
For the segmentation and feature estimation part it is interesting to investigate runtime behavior with increasing region sizes. Since the aim is to obtain an indicator of complexity with increasing amount of input data, considerations on the development platform are sufficient. For the smartphone, performance

with the largest available pill as well as a pill of average size may serve as an indicator for the performance in a real scenario.

All runtime experiments for the laptop are carried out on *set DF*, since better estimates may be expected with a larger amount of input data. Throughout these experiments, optimal system parameters are used (shape PGH: D12-A12, color LUT: $s_{LUT} = 36$). The StbES application was modified for mobile devices to use only fixed-point arithmetic in the entire recognition procedure (see Section 5.4). This means that the runtime on the laptop is comparatively slow, because all operations are carried out with integer arithmetic. With an appropriate scaling in terms of clock frequency, statistics may also hold for smartphones up to a certain degree. All obtained results are divided according to the necessary steps within the pill recognition application.
In Table 13, detailed results on the laptop are provided, when the entire set DF is processed. It must be noted that these values are given with respect to the

Resolution	**Task**	**Max.**	**Min.**	**Mean.**
Laptop	**overall**	**293**	**139**	**217**
640 x 480	preparation	2	1	1
	segmentation	292	138	216
	shape estimation	3	1	2
	size estimation	17	1	6
	color estimation	56	2	17

Table 13: Runtime evaluation on *set DF* (laptop): results with optimal settings (all times in ms, rounded)

entire set. This means, that the runtimes of individual steps do not add up to the given overall time.

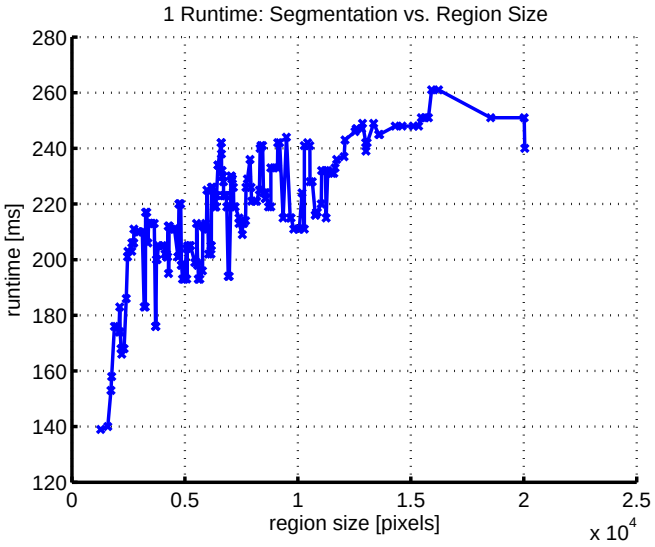

Figure 50: Runtime evaluation (laptop, 640 x 480 pixels): segmentation

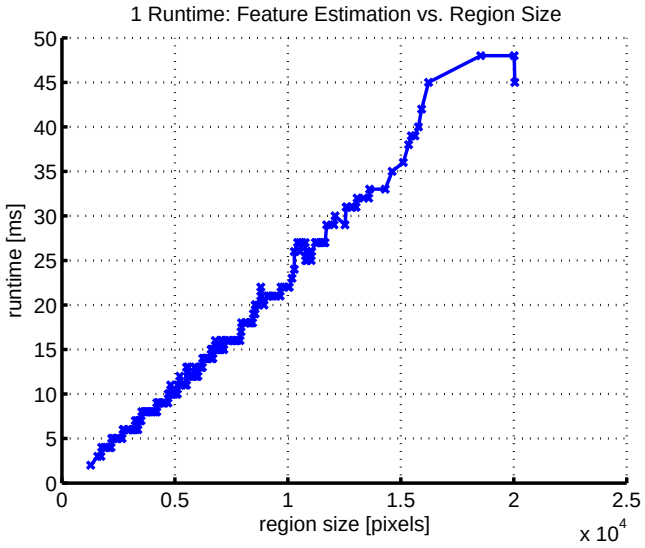

Figure 51: Runtime evaluation (laptop, 640 x 480 pixels): feature estimation

The major part of runtime is needed for the segmentation step (see Table 13). The spikes of runtime with ascending region sizes (see Figure 50) may be attributed to size filtering of regions. Although all regions that lie outside a given range of region sizes are removed, some processing has to take place for them, nonetheless. This runtime may appear as a spike in the diagram.
The runtimes for feature estimation are comparatively low and show a linear dependence on region size (see Figure 51). This means that the application scales well enough so that larger examples still remain computable.

From the results on representative pills a comparison of results for different platforms becomes possible (see Table 14). Obviously the current implementation is about five times slower on the smartphone. In particular, segmentation is a very slow processing step. Increasing region sizes do not overly deteriorate the results, because segmentation is already the dominant step. Computation of the PGH takes a comparatively large relative amount of time (see Figure 52 A and B). Surprisingly, the time for shape classification is less an issue.

Despite limitations in processing speed, the problem may be computed on the mobile test device in a reasonable amount of time. Nonetheless, there is still potential for optimization within the current implementation. Recent smartphones feature more advanced fixed-point processing units with clock frequencies of more than 800 MHz. With such hardware, average processing times can be expected to approach 500ms for images of 640 x 480 pixels, without any measures of optimization. Alternatively, the added processing power could be used to process larger images, which would particularly enhance the accuracy of size estimation.

Description	Task	RT (largest)	RT (average)
Laptop	**overall**	**249**	**228**
640 x 480	preparation	1	1
	segmentation	200	205
	shape features	1	1
	shape class.	1	1
	size estim.	9	6
	color estim.	37	14
Smartphone	**overall**	**1205**	**1114**
640 x 480	preparation	20	18
	segmentation	698	714
	shape features	204	213
	shape class.	6	4
	size estim.	44	29
	color estim.	233	136

Table 14: Runtime evaluation on specific pills (laptop): results for largest pill in set DF as well as an average pill (optimum settings; all times in ms, rounded), RT...runtime

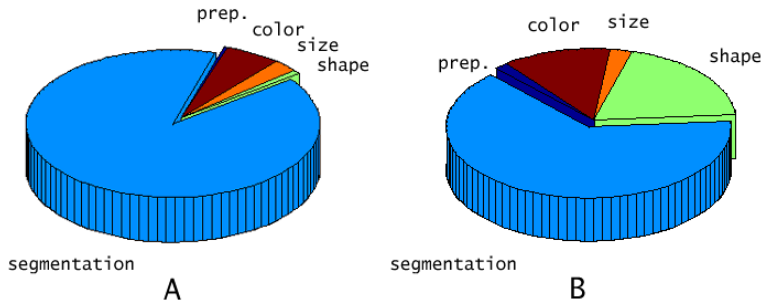

Figure 52: Runtime distribution (pill of average size, 640 x 480 pixels): A...laptop, B...smartphone (scale is not representative)

6.4.5 Pill Retrieval

In the evaluation of pill retrieval performance, a precise identification of samples is necessary. An exact assignment of all samples is not possible, however, since most of them were not delivered with packaging. Although it is clear, that this situation exactly corresponds to a real pill identification scenario, an evaluation is not reasonable without reliable knowledge about the samples.
At the time of this writing, the Identa on-line database suffers from consistency problems. It is not sufficient to issue a single query using length and width information. Certain examples require to issue a second query, where information about the diameter must be provided instead. Otherwise, the system will not present all candidates. This was handled on the server script of the created application and does not hamper evaluation.
Shape classes are not entirely consistent within Identa and additional problems

become evident when querying for colors. As a solution, those samples, which are also listed in Identa, were further filtered according to class consistency in shape and color. This reduces the amount of available samples to only 27 (25%).

Due to this small amount of samples, pill retrieval performance is evaluated in two different ways. First, the created global reference database of manually classified and measured samples, is used as the source of pill information. Although this database only represents a small subset pills, the approach has the advantage that the exact specification of a pill need not be known. Tests can be carried out with all available samples. Additional experiments, which address the descriptive power of a specific feature (or combinations thereof) are also possible.
For the second part, pill retrieval performance is evaluated with Identa using the filtered set of 27 pills. In this case a correction by a human operator is assumed.
For all experiments the pill recognition application is run with with optimal parameters (PGH D12-A12, $s_{LUT} = 36$). The location of the corresponding candidate in the list of answers is taken as an indicator of performance. In other words, a test is performed for each sample we query for, whether the exact match is among the first N candidates or not.

The evaluation of pill retrieval performance on the global reference set is carried out with the images from *set D* (daylight). The global reference set is queried for each test image of *set D* with information from the pill recognition application. In the definition of a query range for length and width, a deviation of $0.7mm$ is assumed. The results of feature estimation are not corrected and may exhibit errors.
Retrieval performance for single features gives information about the descriptive power of a feature (see Figure 53). This means that *size* is the most powerful feature, followed by *color* and *shape* because higher recognition rates are reached with a smaller amount of considered candidates. *Size* and *color* have a steeper slope and give better results faster than the feature *shape*. Performance with the feature *size* reaches a recognition rate of 0.814 ($N = 6$ candidates), when querying for pills using a range defined by the measurements and the expected accuracy in size. Performance of feature *size* may be considerably improved, if results are ordered by the minimum sum of deviations in length and width. With this ordering, a recognition rate of more than 0.99 is reached, when considering 6 candidates from the reference database (see Figure 54).

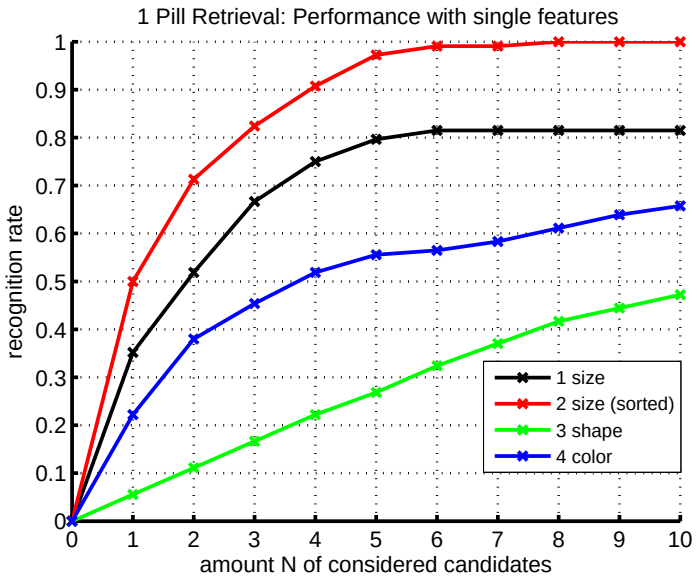

Figure 53: Pill retrieval on the global reference set (single feature, 640 x 480 pixels): results for *set D*

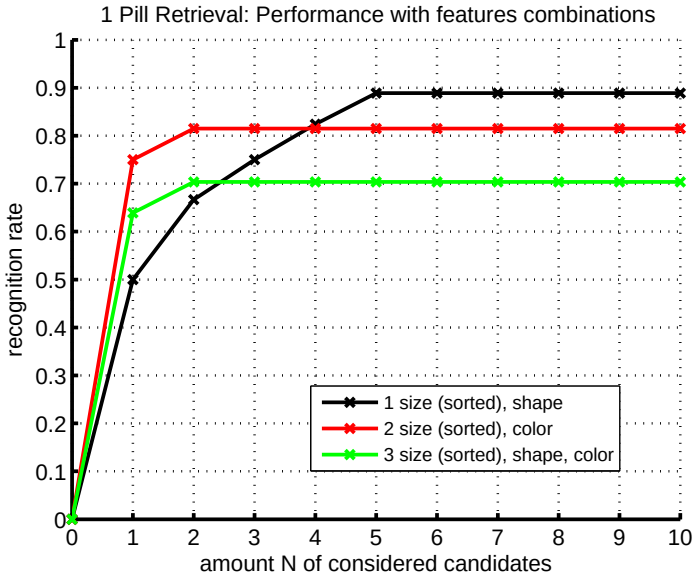

Figure 54: Pill retrieval on the global reference set with size-sorted results (combined features, 640 x 480 pixels): results for *set D*

In a pill retrieval scenario several features are used. In Figure 54, the course of results when using various combinations of features is shown.
From visual inspection it is evident that peak performance drops when using several features. The reason may be mutual reactions of errors from the individual feature estimators, because their results were not corrected. This is particularly evident with the feature color, where the corresponding estimator has a lower recognition rate on the set. Consequently, **human verification of results for *shape* and *color* is necessary** in the process of pill identification. A summary of best results for single features and combinations thereof is presented in Table 15.

Res.	**Size**	**Size (sorted)**	**Shape**	**Color**	RR_{max}	#
640 x 480	x				0.8148	6
		x			**1.0000**	8
			x		0.4722	10
				x	0.6574	10
	x		x		**0.7315**	6
	x			x	0.6389	3
	x		x	x	0.5556	3
		x	x		**0.8889**	5
		x		x	0.8148	2
		x	x	x	0.7037	2

Table 15: Pill retrieval on the global reference set (*set D*, $N = 10$): best results

In the second evaluation of pill retrieval performance, the Identa database is queried with 27 samples using images from *set D*. The results from the individual estimators were corrected, were necessary, and a suitable query range for size was selected, as is the case in a real scenario. It is important to note that we do not have control over the contents of the Identa database. Thus, the results are only valid with respect to the current contents (2009/12). In Table 16 the contents of

Nr.	**Length/Width [mm]**	**Shape**	**Color(s)**	**Position**
1	6.50/6.50	circular	blue	1
2	17.52/6.32	oblong	red/rose	1
3	8.00/5.74	oval	white	10
4	9.65/9.65	circular	rose	21
5	8.03/5.74	oval	blue	1
6	18.97/8.93	oval	yellow	1
7	10.16/10.16	circular	white	-
8	16.29/8.28	oval	rose	-
9	8.13/4.64	oval	blue	17
10	10.22/9.24	special	white	23
11	15.19/6.19	oval	red	1
12	10.20/5.38	oval	rose	1
13	17.80/6.25	oblong	white	4
14	10.09/5.13	oblong	white	14
15	14.25/5.28	oblong	ocher	1
16	15.70/5.75	oblong	blue	1
17	18.55/8.35	special	white	5
18	21.42/7.60	oblong	white	2
19	21.00/7.55	oblong	pink	2
20	8.07/8.07	circular	white	-
21	11.35/11.35	circular	white	-
22	15.74/5.82	oblong	blue/white	5
23	7.11/7.11	circular	rose	20
24	12.16/6.60	oval	white	23
25	9.31/9.31	circular	yellow	6
26	11.69/6.01	oval	beige	9
27	17.15/7.12	oblong	rose	1

Table 16: Pill retrieval with Identa ($N = 25$): best results on *filtered set D* for Identa contents 2009/12

this test set are given, along with the corresponding result (size measurements as obtained with the nonius). The final retrieval performance with Identa is 85.19 percent of all queried examples. On average, $N = 8$ candidates must be inspected to obtain the correct match.

7 Conclusion

7.1 Summary

In this work we present a system that may aid a human operator with the task of identifying pharmaceutical pills in a mobile scenario. The target platform of this system is a conventional smartphone, equipped with a camera and a reasonable amount of memory and computational resources. Following a short introduction and motivation of the work presented here, an overview of existing work is given while keeping the limitations of mobile devices as well as the expected operating conditions in mind. A domain analysis on such objects as well as possible means for identification is carried out to identify the most important aspects of such an identification framework, be it in object segmentation or feature extraction. An algorithm for segmentation of pills from a business card sized marker is presented. Hereafter, relevant properties, such as size, shape and color are chosen to be determined by means of CV. These properties can be used for identification of medical pills within a database. An example of this application interfacing an online-available database for medical matters is presented, followed by an extensive evaluation of the system in terms of algorithm speed and recognition accuracy.

The most difficult subproblem identified during algorithm development is that of segmentation, because pills may be of arbitrary color and the conditions for image acquisition may vary. An efficient method can be applied, when using a specially designed marker-based target. Objects need to be placed on a chessboard background within the target and can be segmented using local adaptive thresholding and morphological operations. This method works for convex objects and is proven to be robust towards changes in lighting and rotation during its experimental evaluation.
Given the object boundary of a pharmaceutical pill, algorithms of different complexity and accuracy are used for feature extraction. Shape estimation is performed by analyzing object contours. A modified Pairwise Geometric Histogram is used as shape descriptor, which is shown to be sufficiently invariant to changes in scale. The subsequent shape matching step can be accomplished efficiently with an approach based on neighbor search using the Euclidean distance. Although we limited the shape estimation algorithm to handle convex shapes only, this is not a general limitation. Thus the approach may also be extended to handle arbitrary shapes. The object size estimation procedure is based on ellipse fitting using rectified region points. Points that make up the convex hull of the contour are projected onto the major and minor axis of the corresponding ellipse. The maximal error in size estimation is bounded by the size of the chessboard pattern embedded on the target. For the estimation of object color, a method for creating a look-up table from a small amount of example colors is employed. A white balancing method using reference measurements on the target is applied to reduce the influence of lighting. Local white point estimates can be drawn and appropriate scaling is carried out. The method is suitable for smaller deviations in lighting and works with arbitrary classes of visually similar colors. For improved robustness, color classes must have sufficient visual distance.

A user interface was designed and implemented that is targeted towards mobile application. It allows an easy application of the procedure and integrates image acquisition, feature estimation and modification, as well as optional functionality to retrieve pill information given a set of extracted features. For the latter, a piece of software is used, that serves as a standardized interface between the application and an online database. This architecture allows to minimize the amount of transferred data with respect to the client. Furthermore, this has the advantage that the interface may be adapted to an arbitrary database with ease.

To prove the usability of the system, its performance is investigated on test data, which represents possible operating conditions for such a system. The obtained results show that the approach is able to work in real-time using commonly available smartphone hardware. Furthermore, the retrieval performance of the system on the exemplarily used Identa database confirms that the system can facilitate the task of mobile pill recognition in a realistic scenario.

7.2 Issues and Future Prospects

As with any system that is designed to fulfill a given task, there are limits. In case of the system at hand, a perfect estimation of parameters is not possible for all operating conditions. Consequently, the final decision on the query parameters is left to the human operator for reasons of safety. This means that manual verification or correction of shape and color will be accomplished by the user. Given a suitable database, the created pill recognition system may simplify as well as accelerate the recognition of pills in a mobile environment by suggesting pill properties and estimating object size more accurately, than what can be obtained with a ruler.

The current system could be enhanced in various ways. A major issue during the construction of the system was the lack of standardization concerning shape and color classes. This also applies to online databases, which suffer from consistency problems. With a general description of these features, more precise training and retrieval would be possible, enhancing the overall performance.
As the quality of the input image is critical in the system at hand, an automatic assessment is conceivable. In a preprocessing step the current video image could be automatically analyzed concerning sharpness, scale and distortion. So, the user can be informed about the suitability of the current image before the actual processing takes place.
The verification of shape and color can be facilitated by presenting exemplary shapes and colors to the user. With standardized feature classes, a considerable increase in the final quality of feature estimation may be possible. The quality of size estimation could be enhanced by using a larger number of correspondences for estimating the homography. Together with an increase of resolution, better results can be expected. Once more, the influence of radial distortion must be evaluated in this case. With such enhancements, runtime optimization is very important, because usability might suffer.
When a suitable and consistent database is available, additional features such as brick lines could be estimated. As the system is always used by a human operator, the advantage is questionable, though. Reasoning about different features such as the type of pill (presentation) from the current result of estimators

can be another possibility to reduce the amount of required input.
As most smartphones feature either a slot for a memory card or have a larger amount of internal memory, all pill information could be stored directly on the mobile device. This would allow the creation of an autonomous system, which can be used regardless of connectivity problems. The actuality of pill information might suffer, however. In this case, an incremental update mechanism can put things right.
The perception of a problem from the point of the developer can be rather different when compared to the target audience. Feedback about the usability and viability of the current system should be obtained from suitable tests involving all types of potential users.

Parts of the system could be used for different tasks. If the constraint of convexity is removed, arbitrary objects can be measured. In this case it might be useful to let the user specify the directions of measurement. In industrial vision the proposed chessboard-based segmentation method could be used as an alternative for setups requiring transmitted light.

Appendices

A Color Spaces

Color models specify a coordinate system and a subspace (color space), where each color is represented by a single point (see [25]). Models differ in choice and quantification of parameters, coverage of the visual spectrum and uniformity. Another distinction is a possible dependency on the device used for sensing or presentation.

One of the first spaces based on a mathematical definition was the *CIE XYZ color space*. Colors may be specified using the so-called tristimulus values X, Y, Z (*CIE chromaticity values*). It is important to note that they are virtual and do not represent the physically observed colors red, green and blue, however. A color is determined by the *trichromatic coefficients* x,y,z then.

$$x = \frac{X}{X+Y+Z} \tag{58}$$

$$y = \frac{Y}{X+Y+Z} \tag{59}$$

$$z = \frac{Z}{X+Y+Z} \tag{60}$$

$$x + y + z = 1 \tag{61}$$

For any wavelength in the visible spectrum, these values may be obtained directly from curves or tables that were generated from experimental data. In contrast to an effect called *metamerism*, two light sources producing the same effect on the observer will have the same tristimulus values X,Y and Z, regardless of their (mixture of) wavelengths.
The model is not device dependent, but transformation into RGB space is a complex non-linear operation. In CIE XYZ, colors of equal perceptual differences may exhibit differing Euclidean distances. Such colors lie on the contours of elliptic areas (*MacAdam Ellipses*) instead (see [37]).

Based upon CIE XYZ are the *uniform* CIE L*a*b* (CIE LAB) and CIE L*u*v* (CIE LUV) spaces. Transformation is carried out from CIE XYZ space with consideration of lighting, specified in CIE XYZ coordinates (often CIE D65 standard light). A uniform color space is a color space in which same-size changes in the color coordinates also correspond to same-size recognizable changes in the visible color tones and color saturation. However, no color space exists that allows an undistorted or unbiased representation. Therefore, uniform means in this sense *more or less uniform*. Uniform color spaces are interesting for several areas of color image processing applications, especially if very similar colors have to be compared. CIE LAB is used for the description of non luminous materials and CIE LUV is usually used for the description of light. CIE LAB is widely used today and may be used for description of all visible colors (see [54]). This space uses a three-dimensional rectangular coordinate system with the quantities L^*, a^*, b^*.

- L^* lightness: scaled approximately according to sensitivity
- a^* red-green axis: negative values denote green colors, positive values denote red colors
- b^* yellow-blue axis: negative values denote blue colors, positive values denote yellow colors

The a^* and b^* axes define color planes, whose brightness is dependent on the vertical axis L^*, on which the achromatic colors are situated.
CIE LCH is a color space that can be obtained by conversion of the CIE LAB rectangular coordinates to cylindrical coordinates. The newly obtained quantities are chroma radius (C, parameter for saturation) and hue angle (H, color of dominant wavelength). The conversion makes the space more suitable for human specification of colors.
Both CIE LUV and CIE LAB spaces are device-independent color spaces. Transformation, however, is computationally expensive and generally involves computation of the CIE XYZ representation as an intermediate step. Despite the fact that CIE LAB and CIE LUV are termed uniform color spaces, research has lead to additional color spaces and color difference metrics that further improve correspondence with human perception over using the Euclidean distance (see Section 2.6.1).

The *RGB model* is a device dependent technical color model that is based on a cartesian coordinate system. Each color appears in its primary components corresponding to long, middle and short wavelengths (for ease of reference called tristimulus values red, green, blue), given as integers. The subspace of interest can be displayed as a cube, where the primary colors are at the corner and the secondary colors (cyan, magenta, yellow) are at the opposite corners. Black is at the origin and white is farthest away, with shades of gray lying on the connecting line. Normalization with respect to intensity can be achieved by dividing each component by the sum of R, G and B and representation of the values using fixed-point or floating-point data types.
The RGB color space is the most applied computer-internal representation of color images (see [37]). Almost all visible colors can be represented by linear combination of the three primaries. For identical objects, differing color values are generated with different sensing or presentation devices since their primary colors in general do not match (device dependence). The process of adjusting color values between different devices uses information in the form of *color profiles* and is called *color management*.
Variant of the RGB color space was introduced in 1996 by the ICC (International Color Consortium) with the aim to create a simple and robust *device independent* color space. The *sRGB color space for the Internet* considers CRT (cathode ray tube) monitors as means of reproduction and daylight (D65) as viewing condition. In an initial step sRGB values are computed as linear combinations of the CIE XYZ values. In the next step the gamma is adjusted (non-linear correction of color values to allow linear perception) in accordance with CRT monitors and finally the values are scaled into an integer representation (8 bits per channel). When no additional information is available, one can assume that an 8 bit per channel image is in sRGB format. Besides, monitors that do not follow the sRGB standard internally, employ additional effort to

comply externally with it. A detailed description, including the standard viewing environment is available in [59].

In the *HSI color space* hue, saturation and intensity are used as coordinate axes [37]. This color space is well suited for the processing of color images and for visual definitions of color. The hue H of a color characterizes the dominant wavelength contained in the color and may be denoted as an angle in degrees. The saturation S of the color is a measurement of color purity and is denoted as parameter of the radius. This parameter is dependent on the number of wavelengths that contribute to the current color perception. The wider the range of the wavelengths, the lower the purity of the color. The more narrow the range of the wavelengths, the higher the purity of the color. The extreme case $S = 1$ occurs for a pure color and the extreme case $S = 0$ for an achromatic color. The intensity I of the color corresponds to relative brightness (in the sense of a gray-level image) and is plotted along the major axis. The extreme case $I = 0$ corresponds to the color black.
One of the advantages of the HSI color space is the separation of chromatic and achromatic information. On the other hand, this model is device dependent in general and the transformation into RGB space is non-linear. A related space is HSV (hue, saturation, value), which may be obtained by projecting the RGB unit cube along the diagonals of white to black and has the form of a single upside-down hexacone.

In *YUV space*[34], color information U,V is separated from brightness information Y. Color information is computed as scaled difference between gamma corrected blue (for quantity U) and red (for quantity V) components and brightness. It is worth noting that the quantity for brightness should be termed luma (Y') and does not match the CIE definition of brightness (see [47]). Due to the fact that human perception is more sensitive to changes in brightness, the bit depth for U and V components may be reduced. The YUV has better properties concerning uniformity than the RGB space but is device dependent as well.
A related space is YC_bC_r for video devices, which uses different offsets and scaling factors, however. The transform from RGB space may be approximated using integer math and as such can be computed efficiently.

B Integral Images

Summed area tables (SAT) or integral images were applied by Crow in 1984 for the purpose of mip mapping (see [12]). A prominent application is found in the real time object detection framework by Viola and Jones (see [66]).
In principle, the purpose of an integral image f_{ii} is to make the computation of sums in a rectangular grid more efficient. It consists of entries which are the sum of all pixel values above and to the left of an image f.

$$f_{ii}(x, y) = \sum_{x' \leq x, y' \leq y} f(x', y') \tag{62}$$

[34] `http://en.wikipedia.org/wiki/YUV`

Computation can be carried in a single pass over the image by using cumulative row sums s_r.

$$s_r(x, y) = s_r(x, y - 1) + f(x, y) \tag{63}$$

$$f_{ii}(x, y) = f_{ii}(x - 1, y) + s_r(x, y) \tag{64}$$

As a result, the sum of pixel values within a rectangular area can be computed with constant computational effort, independent of the window size (see Figure 55). Shafait et al. use integral images and square integral images to compute

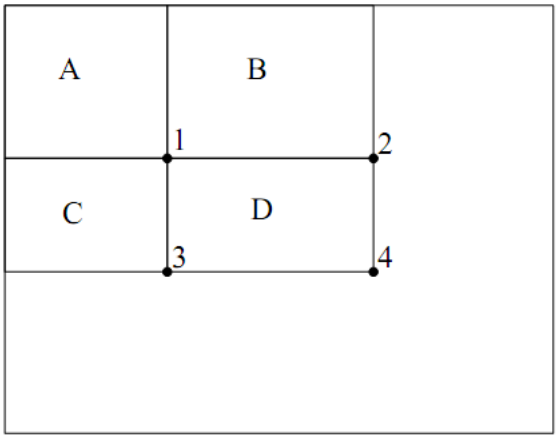

Figure 55: Efficient Accumulation: The sum within the rectangle D can be computed as $4 - (2 + 3) + 1$. (taken from [66])

local means and variances (see [53]). The result is a very efficient implementation of the Sauvola method for local adaptive thresholding (see [50]).

C Calculation of the Covariance Matrix

The following considerations are based upon treatment of measurements as random quantities, which allows to analyze them by means of statistics. In the following, X and Y denote random variables. These are functions, that map the outcome of a random experiment to numbers.

The mean $\overline{x}$ of a set of one-dimensional samples taken from a population may be computed as

$$\overline{x} = \frac{1}{K} \sum_{k=1}^{k=K} x_k, \tag{65}$$

where x_k is a single example.

Standard deviation s is a measure of the spread around the mean and may be computed by geometric averaging of the distance of each sample from the mean value. The variance *var* is a related measure that can be obtained by squaring the standard deviation. The approximation of the variance *var* from

K samples of a population is as follows:

$$var(X) \approx \frac{1}{K-1} \sum_{k=1}^{K} (x_k - \overline{x})(x_k - \overline{x}) \tag{66}$$

These concepts may be extended to higher-dimensional data as well. In the following considerations, column vectors are assumed. Starting with two dimensions (random variables), the covariance *cov* may be computed, which is a measure of change in one dimension with respect to another dimension (correlation). An approximation may be computed as

$$cov(X,Y) \approx \frac{1}{K-1} \sum_{k=1}^{K} (x_k - \overline{x})(y_k - \overline{y}). \tag{67}$$

For an arbitrary amount of dimensions (random variables), all covariances of **two** random variables at a time may be aggregated into the covariance matrix $\mathbf{C_x}$, which is defined by the expression

$$\mathbf{C_x} = \mathbf{E}\{(\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}}\}, \tag{68}$$

where $\mathbf{x}$ is used to denote a sample of arbitrary dimension, $\overline{\mathbf{x}}_\mathbf{x}$ is a vector of corresponding mean values and $E\{.\}$ is the expectation operator (see [25]). Since $cov(A,B) = cov(B,A)$, the matrix is symmetrical along the main diagonal. It may be approximated by evaluation of the following expression:

$$\mathbf{C_x} \approx \frac{\mathbf{1}}{\mathbf{K}} \sum_{\mathbf{k=1}}^{\mathbf{K}} \mathbf{x_k x_k^T} - \overline{\mathbf{xx}}^{\mathbf{T}} \tag{69}$$

References

[1] R. Adelmann, M. Langheinrich, and C. Floerkemeier. A Toolkit for Bar Code Recognition and Resolving on Camera Phones - Jump Starting the Internet of Things. In *Proceedings of the workshop on Mobile and Embedded Interactive Systems (MEIS'06). GI Lecture Notes in Informatics Series (LNI)*, Dresden, Germany, 2006.

[2] A.P. Ashbrook, N.A. Thacker, P.I. Rockett, and C.I. Brown. Robust Recognition of Scaled Shapes using Pairwise Geometric Histograms. In *Proceedings of British Machine Vision Conference*, pages 503–512, 1995.

[3] S. Belongie and J. Malik. Matching with Shape Contexts. In *Proceedings. of IEEE Workshop on Content-based Access of Image and Video Libraries, 2000*, pages 20–26, 2000.

[4] E. Borenstein, E. Sharon, and S. Ullman. Combining Top-Down and Bottom-Up Segmentation. *Conference on Computer Vision and Pattern Recognition*, page 46, 2004.

[5] J. E. Bresenham. Algorithm for Computer Control of a Digital Plotter. *IBM Systems Journal*, 4(1):25–30, 1 1965.

[6] Canalys. Canalys Research Release 2009/081: Smart Phones Defy Slowdown. `http://www.canalys.com/pr/2009/r2009081.htm`, 2009. last visit: 2009/12/27.

[7] J. F. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[8] Fu Chang, Chun-Jen Chen, and Chi-Jen Lu. A Linear-Time Component-Labeling Algorithm using Contour Tracing Technique. *Computer Vision Image Underst.*, 93(2):206–220, 2004.

[9] C. M. Christoudias, B. Georgescu, and P. Meer. Synergism in Low Level Vision. In *Proceedings of the 16th International Conference on Pattern Recognition*, pages 150–156, 2002.

[10] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[11] A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. PhD thesis, University of Oxford, Robotics Research Group, Department of Engineering Science, 1999.

[12] F. C. Crow. Summed-Area Tables for Texture Mapping. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 207–212. ACM, 1984.

[13] I. Dahm, S. Deutsch, M. Hebbel, and A. Osterhues. Robust Color Classification for Robot Soccer. In *International RoboCup Symposium*, Padua, Italy, 2003.

[14] M. Donoser, H. Bischof, and M. Wiltsche. Color Blob Segmentation by MSER Analysis. In *IEEE International Conference on Image Processing*, pages 757–760, 2006.

[15] R. O. Duda and P. E. Hart. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Comm. ACM*, 15(1):1115, 1972.

[16] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. John Wiley & Sons, New York, USA, 2001.

[17] Medical Economics. *Physician's Desk Reference*. Medical Economics Company, Oradell, N.J, 1998.

[18] C.-J. Estler and H. Schmidt. *Pharmakologie und Toxikologie*. Schattauer Verlag, Stuttgart, Germany, 6th edition, 2007.

[19] A.C. Evans, N.A. Thacker, and J.E.W. Mayhew. Pairwise Represenations of Shape. In *Proceedings of the 11th International Conference on Pattern Recognition , Computer Vision and Applications (Conference A)*, pages 133–136. Los Alamitos, Calif. u.a. IEEE Computer Soc. Pr., 1992.

[20] T. Fawcett. ROC Graphs: Notes and Practical Considerations for Researchers. Technical report, HP Laboratories, 2004.

[21] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient Graph-Based Image Segmentation. *International Journal of Computer Vision*, 59(2), 2004.

[22] G. D. Finlayson, S. D. Hordley, and P. M. Hubel. Color by Correlation: A Simple, Unifying Framework for Color Constancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1209–1221, 2001.

[23] Paul Föckler, Thomas Zeidler, Benjamin Brombach, Erich Bruns, and Oliver Bimber. PhoneGuide: Museum Guidance supported by On-Device Oject Recognition on Mobile Phones. In *MUM '05: Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*, pages 3–10, New York, NY, USA, 2005. ACM.

[24] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 2005.

[25] R.l C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 3rd edition, 2008.

[26] R.L. Graham. An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set. *Information Processing Letters*, 1(4):132–133, 1972.

[27] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2008.

[28] G. Heinisch and W. Gerold. *Arzneimitel Schnellerkennung*. Österreichische Apotheker Verlagsgesellschaft m.b.H., Vienna, Austria, 2000.

[29] H. Hse and A. R. Newton. Sketched Symbol Recognition using Zernike Moments. Technical Report UCB/ERL M03/49, EECS Department, University of California, Berkeley, 2003.

[30] M.-K. Hu. Visual Pattern Recognition by Moment Invariants. *IEEE Transactions on Information Theory*, 8(2):179–187, 1962.

[31] J. Iivarinen, M. Peura, J. Särelä, and A. A. Visa. Comparison of Combined Shape Descriptors for Irregular Objects. In *Proceedings of British Machine Vision Conference*, pages 430–439. Colchester : University of Essex, 1997.

[32] J. Iivarinen and A. Visa. Shape Recognition of Irregular Objects. In *Proceedings of SPIE 2904 Intelligent Robots and Computer Vision XV: Algorithms, Techniques, Active Vision, and Materials Handling*, pages 25–32, 1996.

[33] V. Ivanchenko, J. Coughlan, and H.Y. Shen. Detecting and Locating Crosswalks using a Camera Phone. In *Embedded Computer Vision*, pages 1–8, 2008.

[34] B. Kisacanin, S. Bhattacharyya, and S. Chai. *Embedded Computer Vision*. Springer, London, UK, 2009.

[35] S. Kolski., editor. *Mobile Robots: Perception & Navigation*, pages 149–164. Pro Literatur Verlag, Germany ARS, Austria, 2007. Accurate Color Classification and Segmentation for Mobile Robots.

[36] A. Koschan and M. Abidi. Detection and Classification of Edges in Color Images. *Signal Processing Magazine (Special Issue on Color Image Processing)*, 22(1):64–73, 2005.

[37] A. Koschan and M. Abidi. *Digital Color Image Processing*. Wiley-Interscience, Hoboken, New Jersey, USA, 2008.

[38] S. B. Kotsiantis. Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, 31:249–268, 2007.

[39] M. R. Luo, C. Minchew, P. Kenyon, and G. Cui. Verification of CIEDE2000 using Industrial Data. In *Proceedings of Association Internationale de la Couleur Color and Paints (Interim Meeting of the International Color Association)*, pages 97–102, 2004.

[40] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *Proceedings of British Machine Vision Conference*, volume 1, pages 384–393, 2002.

[41] A. Mulloni, D. Wagner, and D. Schmalstieg. Mobility and Social Interaction as Core Gameplay Elements in Multi-Player Augmented Reality. In *DIMEA '08: Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, pages 472–478, New York, NY, USA, 2008. ACM.

[42] D. Nister and H. Stewenius. Linear Time Maximally Stable Extremal Regions. In *Proceedings of European Conference on Computer Vision*, pages 183–196, 2008.

[43] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[44] R.A. Palma-Amestoy, P.A. Guerrero, P.A. Vallejos, and J. Ruiz-del Solar. Context-Dependent Color Segmentation for Aibo Robots. In *IEEE 3rd Latin American Robotics Symposium*, pages 128–136, 2006.

[45] K. Pearson. On Lines and Planes of Closest Fit to Systems of Points in Space. *Philosophical Magazine*, 2(6):559–572, 1901.

[46] M. Peura and J. Iivarinen. Efficiency of Simple Shape Descriptors. In *Advances in Visual Form Analysis*, pages 443–451. World Scientific, 1997.

[47] C. Poynton. YUV and Luminance Considered Harmful. `http://www.poynton.com/PDFs/YUV_and_luminance_harmful.pdf`, 2008.

[48] J.R. Quinlan. *C4.5 - Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, USA, 1993.

[49] M. Rohs. Real-World Interaction with Camera-Phones. In *2nd International Symposium on Ubiquitous Computing Systems (UCS 2004*, pages 74–89. Springer, 2004.

[50] J. Sauvola and M. Pietikinen. Adaptive Document Image Binarization. *Pattern Recognition*, 33:225–236, 2000.

[51] C. Scheering and A. Knoll. Fast Colour Image Segmentation using a Pre-Clustered Chromaticity-Plane. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 4:3145, 1997.

[52] D. Schmalstieg and D. Wagner. Experiences with Handheld Augmented Reality. In *ISMAR '07: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–13. IEEE Computer Society, 2007.

[53] F. Shafait, D. Keysers, and T. Breuel. Efficient Implementation of Local Adaptive Thresholding Techniques Using Integral Images. In *Proceedings of the 15th Document Recognition and Retrieval Conference (DRR-2008), January 26-31, San Jose, CA, USA*, volume 6815. SPIE, 1 2008.

[54] A. Sharma. *Understanding Color Management*. Delmar Learning (Thomson Learning Inc.), Clifton Park, NY, USA, 2004.

[55] G. Sharma. *Digital Color Imaging Handbook*. CRC PRESS, Florida, USA, 2003.

[56] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[57] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. PWS Publ., Pacific Grove, Calif., USA, 2nd edition, 1999.

[58] C. Steger. On the Calculation of Arbitrary Moments of Polygons. Technical report, Forschungsgruppe Bildverstehen, Informatik IX, Tu München, 1996.

[59] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta. A Standard Default Color Space for the Internet - sRGB. `http://www.w3.org/Graphics/Color/sRGB`, 1996. last visit: 2009/08/12.

[60] S. Süsstrunk, J. Holm, and G. D. Finlayson. Chromatic Adaptation Performance of Different RGB Sensors. *SPIE (Electronic Imaging)*, 2001.

[61] R. Szeliski. Image Alignment and Stitching: A Tutorial. *Found. Trends. Comput. Graph. Vis.*, 2(1):1–104, 2006.

[62] D.-C. Tseng and C.-H. Chang. Color Segmentation using Perceptual Attributes. In *Proceedings of the 11th International Conference on Pattern Recognition (Conference C: Image, Speech and Signal Analysis)*, pages 228–231, 1992.

[63] L. Vandevenne. Lode's Computer Graphics Tutorial: Flood Fill Algorithm with Stack. `http://www.student.kuleuven.be/~m0216922/CG/floodfill.html`, 2004. last visit: 2009/12/21.

[64] J. A. S. Viggiano. Comparison of the Accuracy of Different White Balancing Options as Quantified by their Color Constancy. In *Proceedings of SPIE*, volume 5301, pages 323–333. SPIE, 2004.

[65] M. VIK. Industrial Colour Difference Evaluation: LCAM Textile Data. In *Proceedings of Association Internationale de la Couleur Color and Paints (Interim Meeting of the International Color Association)*, pages 138–142, 2004.

[66] P. Viola and M. Jones. Robust Real-Time Object Detection. In *International Journal of Computer Vision*, 2001.

[67] S. Wijewickrema and A. P. Paplinski. Principal Component Analysis for the Approximation of a Fruit as an Ellipse, 2004.

[68] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd edition, 2005.

[69] J. R. Wootton, V. V. Reznack, and G. Hobson. US Patent 6535637 - Pharmaceutical Pill Recognition and Verification system. `http://www.patentstorm.us/patents/6535637.html`, 2003. last visit: 2009/08/18.

[70] C. T. Zahn and R. Z. Roskies. Fourier Descriptors for Plane Closed Curves. *IEEE Transactions on Computers*, c-21(3):269–281, March 1972.

[71] P. Zhu and P. M. Chirlian. On Critical Point Detection of Digital Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):737–748, 1995.