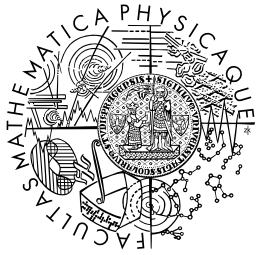


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

DOCTORAL THESIS

Cyril Höschl

Advanced Moment-Based Methods for Image Analysis

Institute of Information Theory and Automation,
the Czech Academy of Sciences

Supervisor of the doctoral thesis:	Prof. Ing. Jan Flusser, DrSc.
Study programme:	Computer Science
Study branch:	Computer Graphics and Image Analysis

Prague 2017

I declare that I carried out this doctoral thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

Prague, 6th June 2017 signature of the author

Práce: Pokročilé momentové metody pro analýzu obrazu

Autor: Cyril Höschl

Katedra: Ústav teorie informace a automatizace, Akademie věd České republiky

Vedoucí disertační práce: Prof. Ing. Jan Flusser, DrSc., Ústav teorie informace a automatizace, Akademie věd České republiky

Abstrakt: Tato disertace rozvíjí pokročilé metody analýzy obrazu založené na obrazových momentech. Zaměřujeme se především na návrh rychlých algoritmů pro počítání momentů v 2D i 3D a vytvoření nových příznaků, které jsou tolerantní ke Gaussovskému rozmazání, resp. zašumění obrazu. Práce se skládá z úvodu do problematiky a čtyř článků. První článek poskytuje přehledovou studii o metodách obdélníkové dekompozice binárních obrázků v 2D; rozklady mj. urychlují počítání momentů. Součástí studie jsou i implemetnace algoritmů vč. optimálního, který existuje v 2D v polynomiální složitosti a je prakticky dosažitelný. Druhý článek se soustředí na dekompozici 3D binárních objektů do kvádrů. Na rozdíl od 2D je v 3D otázka optimálního rozkladu NP-úplný problém a není známo, že by existoval efektivní způsob jeho řešení. V článku navrhujeme nový sub-optimální algoritmus, který pracuje v polynomiálním čase a na experimentální databázi ukazujeme, že dává statisticky významně lepší výsledky, než nejlepší známé heuristiky. Další dva články se soustřeďují na příznaky invariantní ke Gaussovskému rozmazání a zašumění obrazu. Třetí článek představuje invarianty založené na projekčních operátorech ve Fourierově doméně, což zvyšuje především jejich rozlišovací schopnost. Poslední článek představuje robustní příznaky histogramu obrázku. Metoda je tolerantní vůči Gaussovskému zašumění původního obrazu a na rozsáhlých experimentech ukazujeme, že významně převyšuje běžně používané metody.

Klíčová slova: obrazové momenty, momentové invarianty, rozklad na obdelníky, zašuměný obraz, vyhledávání obrazu podle obsahu

Title: Advanced Moment-Based Methods for Image Analysis

Author: Cyril Höschl

Department: Institute of Information Theory and Automation,
the Czech Academy of Sciences

Supervisor: Prof. Ing. Jan Flusser, DrSc., Institute of Information Theory and Automation, the Czech Academy of Sciences

Abstract: The Thesis consists of an introduction and four papers that contribute to the research of image moments and moment invariants. The first two papers focus on rectangular decomposition algorithms that rapidly speed up the moment calculations. The other two papers present a design of new moment invariants. We present a comparative study of cutting edge methods for the decomposition of 2D binary images, including original implementations of all the methods. For 3D binary images, finding the optimal decomposition is an NP-complete problem, hence a polynomial-time heuristic needs to be developed. We propose a sub-optimal algorithm that outperforms other state of the art approximations. Additionally, we propose a new form of blur invariants that are derived by means of projection operators in a Fourier domain, which improves mainly the discrimination power of the features. Furthermore, we propose new moment-based features that are tolerant to additive Gaussian image noise and we show by extensive image retrieval experiments that the proposed features are robust and outperform other commonly used methods.

Keywords: image moments, moment invariants, rectangular decomposition, noisy image, content based image retrieval

I would like to express my gratitude especially to my supervisor, Prof. Ing. Jan Flusser, DrSc., for his mentoring and patient guidance not only with the preparation of this Thesis, but also for his support during the studies at the Faculty of Mathematics and Physics and Institute of Information Theory and Automation (IITA). Additionally, I would like to thank all my fellow colleagues at the Image Processing Department at IITA for their contribution to the friendly and inspiring atmosphere there. Also I would like to thank my friend Benjamin Fistein for his repeated help with grammar checks. Last but not least, I would like to thank my wife Eva and my children Ellen and Dan not only for posing as models in two illustrative photos (Figs. 2 and 3), but mainly for their understanding and joy they bring.

Acknowledgement: The author thanks the Czech Science Foundation for financial support under the Grant No. GA15-16928S.

Contents

1. Introduction

Every day, humans process enormous amounts of visual information. The vast majority of inputs we receive is of a visual kind. Face recognition is fundamental for social interactions, character recognition is necessary for reading texts and the recognition of visual objects in general is essential for understanding and navigating in this world. The image is a very powerful information medium that serves not only as a source of information, but also as a communication interface between machines and humans. Recent trends show an increased popularity in feeding computers with a visual input rather than a traditionally encoded one. To name a few examples, today's mobile applications can read invoice information through QR scans, label objects in photos, recognize wine bottles to provide instant reviews, and many more. A basic digital image captured by an ordinary smartphone contains a similar amount of information as hundreds of text pages. Thanks to the valuable information contained in digital images, many application areas such as medicine, robot vision, astronomy, surveillance or remote sensing urges a strong need for effective digital image analysis methods.

1.1. Digital image analysis by a computer

The process of digital image analysis consists of multiple steps in which the input image is transformed into a final symbolic description of the image or to a decision result. A typical example is the input image of a family photo and the recognition analysis identifying individual faces on the photo against a given database of people. Other examples include the identification of a possible tumor in PET images, localization and recognition of vehicle registration plates, reading road signs, recognition of a handwritten text, etc.

The workflow of the image analysis typically consists of the steps illustrated in Fig. 1. Fig. 2 shows the typical process of an image analysis on an illustrative example of a family photo.

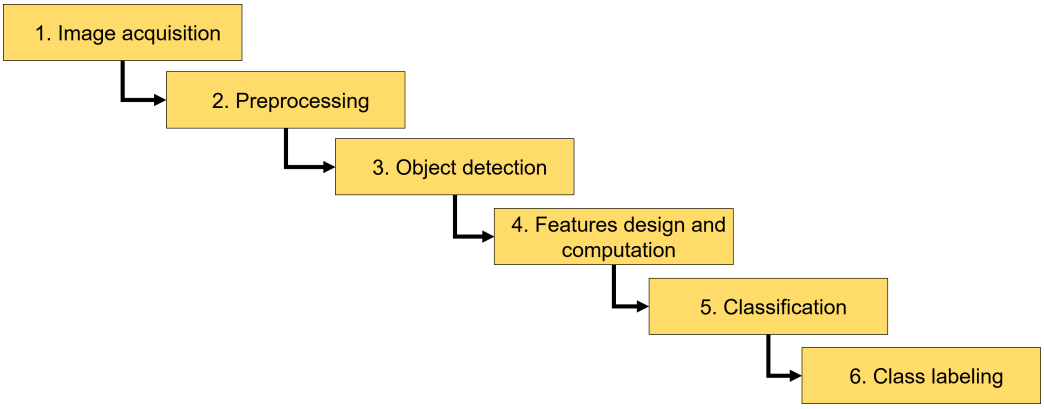


Figure 1: The scheme of image analysis workflow.

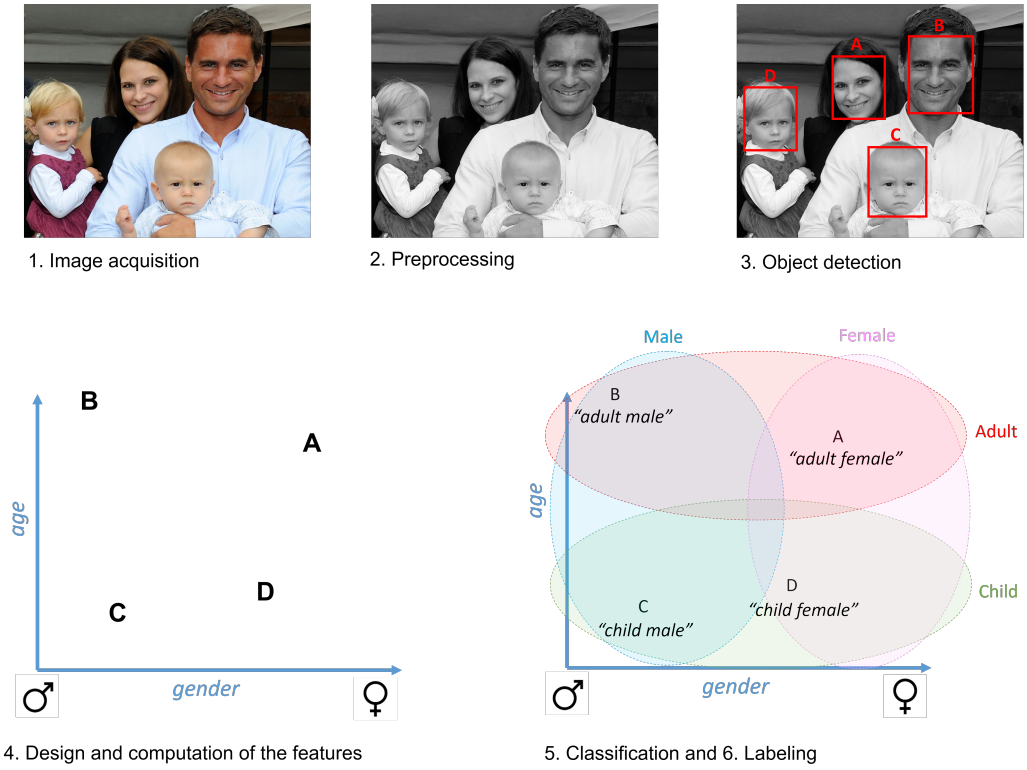


Figure 2: An illustrative example of an analysis of a family photo. The individuals are categorized by their age and gender.

The first three steps (image acquisition, preprocessing and object detection) are exhaustively covered in several image processing books [1, 2, 3, 4, 5], recent monographs [6, 7, 8] and thousands of research papers. In the following paragraphs, we will describe them only very briefly. This Thesis mainly focuses on methods in the fourth step: the feature design. We focus on a special class of features, which are based on image moments. The fifth step of the classification will also be mentioned very briefly as its detailed explanation is beyond the scope of this text.

1.1.1. Image acquisition

In image acquisition, the main theoretical questions are how to choose the sampling scheme, the sampling frequency and the number of the quantization levels such that the artifacts caused by aliasing, moire, and quantization noise do not degrade the image much while keeping the image size reasonably low (there are of course also many technical questions about the appropriate choice of camera and spectral band, objective, memory card, transmission line, storage format, etc., which we do not discuss here). Since real imaging systems as well as imaging conditions are usually imperfect, the acquired image represents only a degraded version of the original scene. Various kinds of degradations (geometric as well as graylevel/color) are introduced into the image during the acquisition process by such factors as imaging geometry, lens aberration, wrong focus, motion of the scene, systematic and random sensor errors, noise, etc. (see Fig. 3 for the general scheme and an illustrative example). The removal or at least suppression of these degradations is a subject of image preprocessing. Historically, image preprocessing was one of the first topics systematically studied in digital image processing (already in the very first monograph [9] there was a chapter devoted to this topic), because even the simple preprocessing methods were able to enhance the visual quality of the images and were feasible on old computers. The first two

Figure 3: Image acquisition process with degradations.

steps, image acquisition and preprocessing, are often categorized in literature into low-level processing. The characteristic feature of low-level methods is that both their input and output are digital images. On the contrary, in high-level processing, the input is a digital image (often an output of some preprocessing) while the output is symbolic (i.e. high-level) information, such as the coordinates of the detected objects, the list of boundary pixels, etc.

1.1.2. Object detection

Object detection is a typical example of high-level processing. The goal is to localize the objects of interest in the image and separate (segment) them from the other objects and from the background. Hundreds of segmentation methods have been described in the literature. Some of them are universal, but most of them were designed for specific families of objects such as char-

acters, logos, cars, faces, human silhouettes, roads, etc. A good basic survey of object detection and segmentation methods can be found in [5] and in the references therein.

1.1.3. Feature definition

Feature definition and computing are probably the most challenging parts of image analysis. The features should provide an accurate and unambiguous quantitative description of the objects. The feature values are elements of the feature space, which should be of low dimensionality for the sake of an efficient computation. The design of the features is highly dependent on the type of objects, on the conditions under which the images have been acquired, on the type and the quality of preprocessing, and on the application area. There is no unique "optimal" solution.

1.1.4. Classification/recognition

The fifth and last step in the image analysis pipeline is performed entirely in the feature space. Every unknown object in the image is represented by a point in a (typically low dimensional) feature space. The object will be assigned to the proper class based on the position of the respective point.

The classes can be prepared in advance by samples that create a *training set*. In this case, we talk about *supervised classification*. If no training set is available, classes are formed according to the distribution of the unknown objects in the feature space. Then we talk about *unsupervised classification* or *clustering*.

Unlike the feature design, the classification methods are independent of the nature of the original data or physical meaning of the features. Classification algorithms are intensively covered in the popular Duda-Hart-Stork book [10] and in recent monograph [11].

Classification methods are not exclusively related to image processing. We can find many applications of them in other areas, such as artificial

intelligence, social sciences, statistics, decision making and many other fields that are beyond the scope of this work.

1.2. Object Recognition: Humans vs. computers

Computers are very powerful in solving exact numerical problems, where they are much faster and more precise than people. However, the human brain is much better suited for solving image recognition tasks. For computers and today's algorithms, solving a simple task such as a recognition of individuals in a family photo is a very complex and non-trivial problem. For us, on the other hand, it is an easy task, especially due to the fact that humans employ lifetime experience.

The human brain is also very capable of choosing the proper context of a visual object. For example, if we see a picture of a partly occluded person with a napkin around their neck, holding a fork and a knife above a table, we automatically assume that the person is going to eat something from the plate even though the table is occluded (see Fig. 4a). Our experience has taught us that the picture shows someone enjoying his meal even though it is not necessarily always the case. In this example, people quickly guess the right context of dining by assigning a high prior probability to it. Even the speed of the human's investigation of the image does not depend on the number of scenes we have already seen in our life. On the other hand, for computers, the right choice of context and the estimation of the prior probability is a very complex and time-demanding problem.

Computers can in principle solve these tasks, too, but it requires massive learning when the algorithm needs to process an enormous amount of data from a large database, and hence the task is quite slow and memory-demanding (especially the training phase). To overcome this limitation, several recent projects try to provide a substitution for human life experience with a shared knowledge available online (see for example [12]).

Figure 4: Illustrative pictures. a) Human brain automatically assumes the man is about to eat. b) Prior probability of the unknown object is determined according to the known objects nearby.

To avoid the human way of contextual perception, many algorithms analyze isolated objects separately without a broader context or they classify the objects in the neighborhood and then use the relations between the objects as additional clues for the recognition (For example, if we analyze an unknown object and we know that there are airplanes on both sides of the query object, then there is a high prior probability that the investigated object is also an airplane (see Fig. 4b for an illustration).

Another advantage of human brains over traditional computer algorithms is the robustness to various changes, such as a geometric distortion caused by a change of scale, mirroring, rotating, skewing, etc. Computer methods can solve this as well, but they need to involve special functions that are tolerant/insensitive to such modifications. Standard features may rapidly change under a spatial transformation of the image, and hence it may lead to a wrong classification.

1.3. Summary

The short introduction into image analysis, which we gave in this section, illustrates that visual object recognition is a challenging and complex task, which is important in many application areas. It employs advanced

algorithms in all steps of the analysis pipeline.

This work focuses mainly on special types of features for object description and recognition, which are known as image moments. An introduction to moments and a specification of the Thesis goals are provided in the next section.

2. Image Moments

Moments and moment invariants play a very important role as features in invariant recognition. They have been introduced to the image processing community in the early 60's, but an increase of a research interest appeared rapidly in recent years as it is obvious also from a graph (see Fig. 5). In SCOPUS, which is probably the most widely-used publication database, a query on papers with the "image moment" keyword returns more than 18,500 search results and more than 6,500 results on the "moment invariants" search. There has been an overlap of about 3,000 papers, which results in 22,000 relevant papers in total. This large amount of results illustrates the importance that the methods based on image moments play in the area of computer science.

Invariant methods based on image moments are extensively reviewed in the book [13] (we adopted the notation and basic definitions from this book) which is a successor of the moment invariant textbook from 2009 [14]. In 2014, a multi-authored book edited by G. A. Papakostas [15] also appeared on the market and reflects the recent development in some areas, even though it does not cover the whole topic as extensively as the previously mentioned monographs.

2.1. Mathematical preliminaries

Before describing moments in more detail, we first recall some basic definitions that will be useful for understanding the further text.

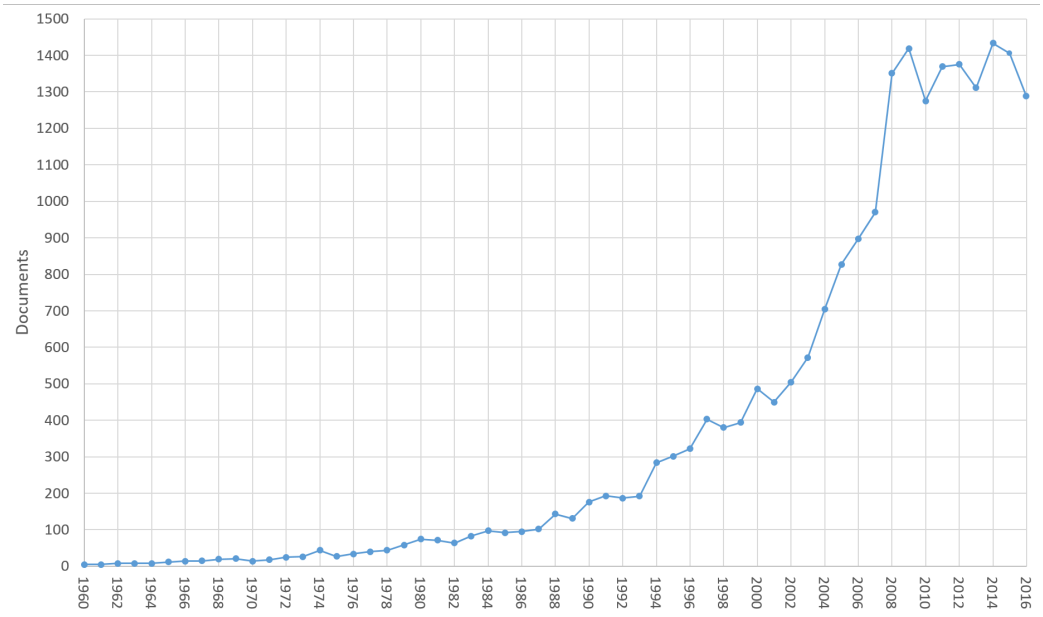


Figure 5: The number of moment-related publications as found in SCOPUS.

Spatial coordinates in the image domain are denoted as $\mathbf{x} = (x_1, x_2, \cdots, x_d)^T$, where d is the dimension of the space. In 2D and 3D domains, if there is no danger of misunderstanding, we sometimes use a simpler (and more common in the literature) notation $\mathbf{x} = (x, y)^T$ and $\mathbf{x} = (x, y, z)^T$, respectively. The supperscript $(\cdots)^T$ means a transposition, so our coordinates are arranged into a column vector.

Definition 1.: By an *image function* (or *image*) we understand any piecewise continuous real function $f(\mathbf{x})$ defined on a compact support $\Omega \subset \mathbb{R}^d$, which has a finite nonzero integral.

According to this definition, our "images" need not be non-negative. The piece-wise continuity and the compact support are assumed to make the operations we are going to apply on the images well defined, which enables a comfortable mathematical treatment without a tedious verification of the existence of the operations in each individual case. From a purely mathematical point of view, these requirements may seem to be too restrictive, because certain operations are well defined on broader classes of functions, such as

integrable or square integrable functions or infinitely supported functions of fast decay. However, these nuances make absolutely no difference from a practical point of view when working with digital images. The non-zero integral is required, because its value will be frequently used as a normalization factor[1].

If the image is mathematically described by the image function, we sometimes speak of a *continuous representation*. Although the continuous representation may not reflect certain properties of digital images, such as sampling and quantization errors, we will adopt the continuous formalism because of its mathematical transparency and simplicity. If the discrete character of the image is substantial (such as in sections 6 and 7 where we explain decomposition algorithms for numerical moment computations), we will work with a *discrete representation* of the image in the form of a finite-extent 2D matrix $\mathbf{f} = (f_{ij})$ or 3D matrix $\mathbf{f} = (f_{ijk})$, which is supposed to be obtained from the continuous representation $f(\mathbf{x})$ by sampling and quantization.

The image function from Definition 1 represents *monochromatic* images (sometimes also called graylevel or scalar images). If f has only two possible values (which are usually encoded as 0 and 1), we speak about a *binary* image. Color images and vector-valued images are represented as a vector image function, each component of which satisfies Definition 1.

Convolution is an operation between two image functions, the result of which is another image function defined as

$$(f * g)(\mathbf{x}) = \int_{\mathbb{R}^d} f(\mathbf{t}) g(\mathbf{x} - \mathbf{t}) \mathrm{d}\mathbf{t} \ . \tag{1}$$

For example, a blurred image can be viewed as a sharp image f that has been convolved with some mask g. If the mask is a Gaussian, then we speak

[1] If we relaxed this assumption, the invariants still could be constructed provided that at least one moment is non-zero.

about a Gaussian blur.

2.2. Moments

Moment is a feature (quantitative measure) that characterizes a given function. It is expressed by a real or complex scalar value. From a mathematical point of view, moments are "projections" of function f onto a polynomial basis.

Let us mention the formal definitions of basic types of moments:

Definition 2.: Let $\{\pi_{\mathbf{p}}(\mathbf{x})\}$ be a d-variable polynomial basis of the space of image functions defined on Ω and let $\mathbf{p} = (p_1, \cdots, p_d)$ be a multi-index of non-negative integers which show the highest power of the respective variables in $\pi_{\mathbf{p}}(\mathbf{x})$. Then the *general moment* $M_{\mathbf{p}}^{(f)}$ of image f is defined as

$$M_{\mathbf{p}}^{(f)} = \int_{\Omega} \pi_{\mathbf{p}}(\mathbf{x}) f(\mathbf{x}) \mathrm{d}\mathbf{x} \ . \tag{2}$$

The number $|\mathbf{p}| = \sum_{k=1}^{d} p_k$ is called the *order* of the moment. We omit the superscript $^{(f)}$ whenever possible without confusion.

Depending on the polynomial basis $\{\pi_{\mathbf{p}}(\mathbf{x})\}$, we recognize various systems of moments. The most common choice is a standard power basis $\pi_{\mathbf{p}}(\mathbf{x}) = \mathbf{x}^{\mathbf{p}}$, which leads to *geometric moments*

$$m_{\mathbf{p}}^{(f)} = \int_{\Omega} \mathbf{x}^{\mathbf{p}} f(\mathbf{x}) \mathrm{d}\mathbf{x} \ . \tag{3}$$

In the literature, one can find various extensions of Definition 2. Some authors allow non-polynomial bases (more precisely, they allow basis functions which are products of a polynomial and some other – usually harmonic – functions) and/or include various scalar factors and weighting functions in the integrand. Some other authors even totally replaced the polynomial basis by some other basis, but still call such features moments – we can find *wavelet moments* [16] and *step-like moments* [17], where wavelets and step-wise functions are used in a combination with harmonic functions instead of polynomials. These modifications broadened the notion of moments, but have not brought any principle differences in moment usage.

2.3. Geometric moments in 2D

In case of 2D images, the choice of the standard power basis $\pi_{pq}(x, y) = x^p y^q$ yields *2D geometric moments*

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) \mathrm{d}x \mathrm{d}y. \tag{4}$$

Geometric moments of low orders have an intuitive meaning – m_{00} is a "mass" of the image (on binary images, m_{00} is an area of the object), m_{10}/m_{00} and m_{01}/m_{00} define the *center of gravity* or *centroid* of the image. Second-order moments m_{20} and m_{02} describe the "distribution of mass" of the image with respect to the coordinate axes.

A characterization of the image by means of geometric moments is complete and unambiguous in the following sense. For any image function, its geometric moments of all orders do exist and are finite. The image function can be exactly reconstructed from the set of all its moments (this assertion is known as the *uniqueness theorem* and holds on an infinitely accurate polynomial approximation of continuous functions thanks to the Weierstrass theorem).

2.4. Other moments

Geometric moments are very attractive thanks to the formal simplicity of the basis functions. This is why many theoretical considerations about moment invariants have been based on them. On the other hand, they have also certain disadvantages, mainly due to the complicated transformations under certain operations (like rotation, translation, etc.). To overcome this drawback, other forms of moments have been invented. For example, *central moments* are well suited to translation transformation, *circular moments* (and their special case *complex moments*) to rotation, *orthogonal moments* lead to the higher numerical precision. Detailed review of various types of moments is covered in the book [13].

2.5. Moment invariants

In the previous sections we have shown the role that the image moments play in the feature design. An important property of well-designed features is an invariance to a certain transformation. Moment invariants are special functions of image moments. They have become important and frequently used shape descriptors. Nowadays, they exist for 2D as well as for 3D objects with a possible extension into arbitrary dimensions in some cases. We have moment invariants for binary, gray-level, color, and even vector-valued images. Concerning the type of invariance, there are moment invariants to similarity and affine object transformations as well as invariants w.r.t. image blurring with certain types of blurring filters.

From a mathematical point of view, invariant I is a functional, which does not change its value under a degradation operator $\mathcal{D}$, i.e. which satisfies the condition $I(f) = I(\mathcal{D}(f))$ for any image function f. The operator $\mathcal{D}$ is supposed to belong to a certain group of transformations but its particular form is not required, which enables working with uncertain information about the deformation. In practice, we usually formulate this requirement as a weaker constraint: $I(f)$ should not be significantly different from $I(\mathcal{D}(f))$. Another desirable property of I, as important as invariance, is *discriminability*. For objects belonging to different classes, I must have significantly different values. Clearly, these two requirements are antagonistic – the broader the invariance, the less discrimination power and vice versa. Choosing a proper trade-off between invariance and discrimination power is a very important and challenging task in feature-based object recognition.

3. Current trends in the moment research

Moment research is a quickly developing area, where the researchers have been studying all aspects of moment-based image analysis. Although an

exact categorization is difficult, we can still identify three main directions in theoretical research, which have attracted major attention in the last years.

3.1. Looking for optimal polynomial basis

Moment invariants are an example of *projection invariants* which are functions of image projections onto proper functional bases. Until now, the choice of the basis was usually done "ad hoc" and then the properties of the invariants were investigated. The recent trend shows an effort to grasp the problem from the other end. First to define the desirable criteria for each particular task and then to construct the basis which optimizes these criteria. In this way, we obtain an image representation which is guaranteed to be optimal in a pre-defined sense, which can be chosen according to the users request.

The major breakthrough is that instead of constructing invariants from the projections on the a priori chosen basis, the new approach is to *adapt and optimize* the basis with respect to the given task and dataset. Hence, it optimizes a cost function reflecting the discrimination power or reconstruction ability of the invariance. This is an important improvement compared to the state of the art, where the basis was usually chosen in advance and the optimization (if ever considered) only selected the "best" projections but did not change the basis.

An optimization of the basis leads to a stronger discrimination power and enables the handling of more general image deformation classes.

3.2. Designing new moment invariants

Moment features invariant to scale, translation and rotation have been known for many decades. Subsequently, the idea of blur invariant functionals appeared in various papers (Flusser *et al.* [18], and Flusser and Suk [19]) as well as several other authors have further developed the theory of blur invariants. Combined invariants to convolution and to rotation were introduced

by Flusser and Zitová [20], who also reported their successful usage in satellite image registration [21] and in camera motion estimation [22]. Combined invariants both to convolution and affine transformation was published by Zhang et al. [23] and Suk and Flusser [24].

A recent important improvement in the moment invariant research is designing new features that are invariant to image noise. New noise invariant features applied on the image histogram are introduced in the paper [25] that is part of this thesis.

3.3. Developing fast algorithms for moment calculations, speed and stability testing

In theory, we mostly work with moments and moment invariants in a continuous domain. In digital image processing, however, all quantities have to be converted from a continuous to a discrete domain; therefore efficient algorithms for a computation of discrete quantities need to be developed. The discrete algorithms sometimes follow their continuous ancestors in a straightforward manner, but sometimes the development of a computationally efficient algorithm requires new inventions. The same holds true for moments and moment invariants.

When speaking about algorithms for moment computation, we generally understand algorithms for binary as well as graylevel images, for geometric and complex moments, for algorithms which speed up the computation and make it more robust to numerical errors and for algorithms, which act in 2D and 3D.

It is not necessary to develop particular algorithms for the calculation of moment invariants. Since all invariants are functions of a small number of moments (relative to the image dimension), the computing complexity of the invariants is determined solely by the complexity of the moment computation. As soon as we have calculated the moments, we can calculate any invariant

in $\mathcal{O}(1)$ time.

Much effort has been spent in recent years to develop efficient algorithms to compute moments. Most authors have focused on binary images because of their importance in practical pattern recognition applications.

The methods for a fast computation of the moments of binary images can be divided into two groups referred to as the *boundary-based methods* and the *decomposition methods*.

In this work, we have focused on algorithms that speed up the computation of moments by decomposing binary images (2D or 3D) into rectangular blocks.

4. Main Goals

The Thesis contributes to image analysis research mainly in the following areas:

- Development of fast algorithms that speed-up moment calculations
- Design of features that are tolerant to Gaussian blur
- Design of robust methods for noisy image retrieval

5. Structure of the Thesis

The Thesis consists of four papers that are attached below. Each paper presents the work that has contributed to achieve the main goals.

The first two papers focus on rectangular decomposition algorithms that help speed-up moment calculations. The last two papers focus on a design of new moment invariants.

The first paper provides a comparative study of state of the art methods for 2D decomposition and discusses their pros and cons. The study includes the original implementation of the optimal 2D decomposition algorithm. We

have also shown how that rectangular decomposition can be used not only to compute moments but also to compress images. To support the decompression stage, we have designed a fast novel algorithm that reconstructs the image outline from a given set of disjoint rectangles.

The second paper proposes a novel heuristic method of effective decomposition in 3D. In contrast to its 2D counterpart, the optimal 3D decomposition is a much harder task and fast approximations are necessary to achieve the goal in the practice. The proposed algorithm outperforms the other cutting edge methods in terms of the number of decomposed blocks and yet its time complexity stays polynomial.

The third paper proposes a new form of blur invariants that are derived by means of projection operators in a Fourier domain (this improves mainly the discrimination power of the features).

The fourth paper utilizes the idea of image blur invariants and proposes a new moment-based feature that is tolerant to image noise (the proposed feature is applied on the image histogram and is invariant to additive white Gaussian noise of the image).

Brief summaries of the attached papers are in the following sections.

6. Decomposition of binary images - A survey and comparison

6.1. Citations

- T. Suk, C. Höschl IV, and J. Flusser, "Decomposition of binary images – a survey and comparison," *Pattern Recognition*, vol. 45, no. 12, pp. 4279–4291, 2012

Shorter version also appeared at

- T. Suk, C. Höschl, and J. Flusser, "Rectangular decomposition of binary images," in *Advanced Concepts for Intelligent Vision Systems:*

14th International Conference ACIVS 2012, (Brno, Czech Republic), pp. 213–224, Springer Berlin Heidelberg, September 2012

6.2. Abstract

We present an overview of the most important methods that decompose an arbitrary binary object into a union of rectangles. We describe a run-length encoding and its generalization, decompositions based on quadtrees, on mathematical morphology, on the distance transform, and a theoretically optimal decomposition based on a maximal matching in bipartite graphs. We compare their performance in image compression, in moment computation and in linear filtering. We show that the choice is always a compromise between the complexity and time/memory consumption. We give advice on how to select an appropriate method in particular cases.

6.3. Main contribution of the paper

This paper provides an overview of the most important methods that decompose a binary 2D image into a disjoint set of rectangles. Image rectangular decomposition has found numerous straightforward applications in image compression methods and formats (RLE, TIFF, BMP and others) and in the calculation of image features (mainly moments and moment invariants) used subsequently for object description and recognition [28, 29, 30, 31, 32, 33, 34, 35]. Other applications may be found in image spatial filtering and restoration and in other areas. For an illustration of various decomposition methods see Fig. 6.

In case of filtering with a binary mask, the time complexity per one pixel depends on the number of rectangles (into which the mask is decomposed) and not the size of the mask itself. If the mask is partitioned into a small number of blocks, it speeds-up the whole filtering process rapidly.

Similarly, computation of image moments and moment features is proportional to the number of rectangles rather than the image size.

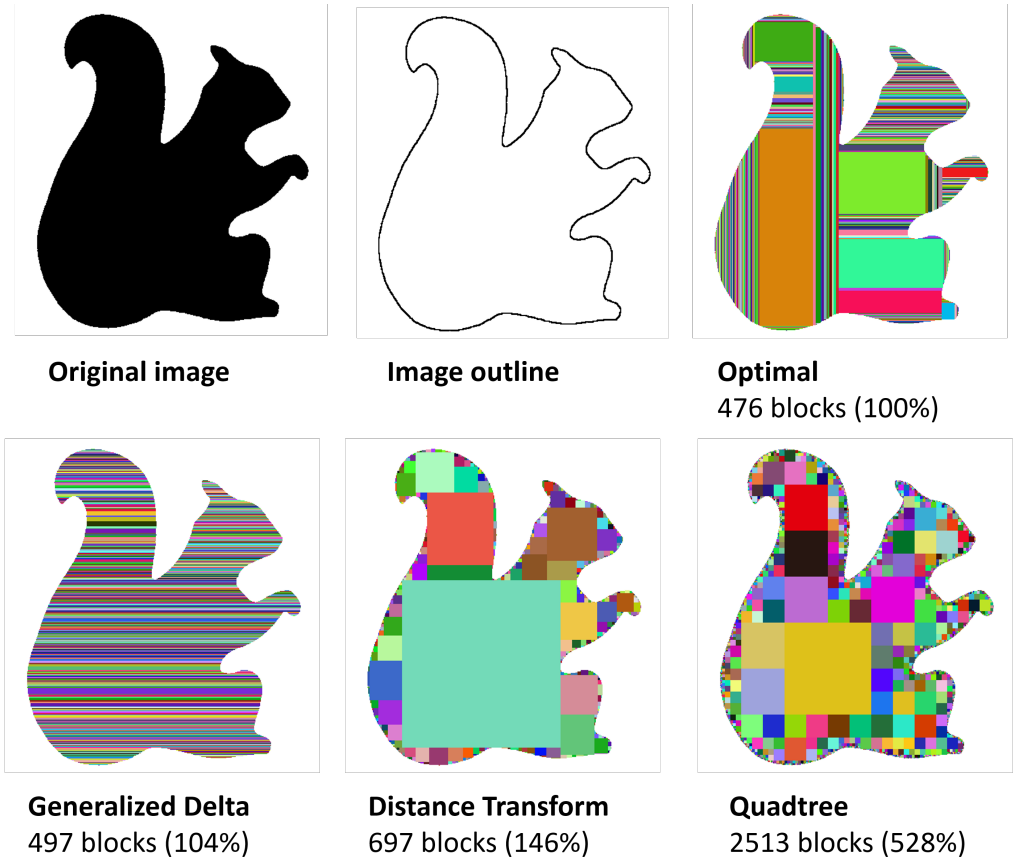


Figure 6: Example of various decomposition methods.

The idea behind the speed-up algorithms is as follows. Let's suppose we have a 2D image, but the principle can also easily be generalized for 3D images. If image B is partitioned into K disjoint blocks, then the moment of the image is the sum of the moments of blocks:

$$m_{pq}^{(B)} = \sum_{k=1}^{K} M_{pq}^{(B_k)}, \tag{5}$$

Thanks to the Newton-Leibnitz formula, the block moment $M_{pq}^{(Bk)}$ can be calculated in $\mathcal{O}(1)$ time. Hence, the image moment $m_{pq}^{(B)}$ can be calculated in $\mathcal{O}(K)$ time. Since the number of blocks is much lower than the image size, the time improvement compared to the trivial calculation is significant.

The crucial task is to find an optimal image partitioning that minimizes the number of disjoint blocks.

The choice of decomposition method is a trade-off between the number of blocks it generates, its computational complexity and space requirements for storing decomposed rectangles. We have shown various methods and discussed their advantages and drawbacks with regards to speed-up and compression purposes.

We have also implemented the optimal method (w.r.t. the number of blocks) that is based on graph algorithms and is solvable efficiently in 2D (in polynomial time). However, some of the other methods yield only slightly worse results and their computation is simpler and faster. See Fig. 7 for an illustration of the steps of the optimal algorithm.

We have shown that the decomposition can also be utilized in image compression where only the rectangular blocks are stored instead of the image pixels. To render the image from a sequence of the stored blocks, we have proposed a new algorithm for a fast and memory-efficient reconstruction of an image outline.

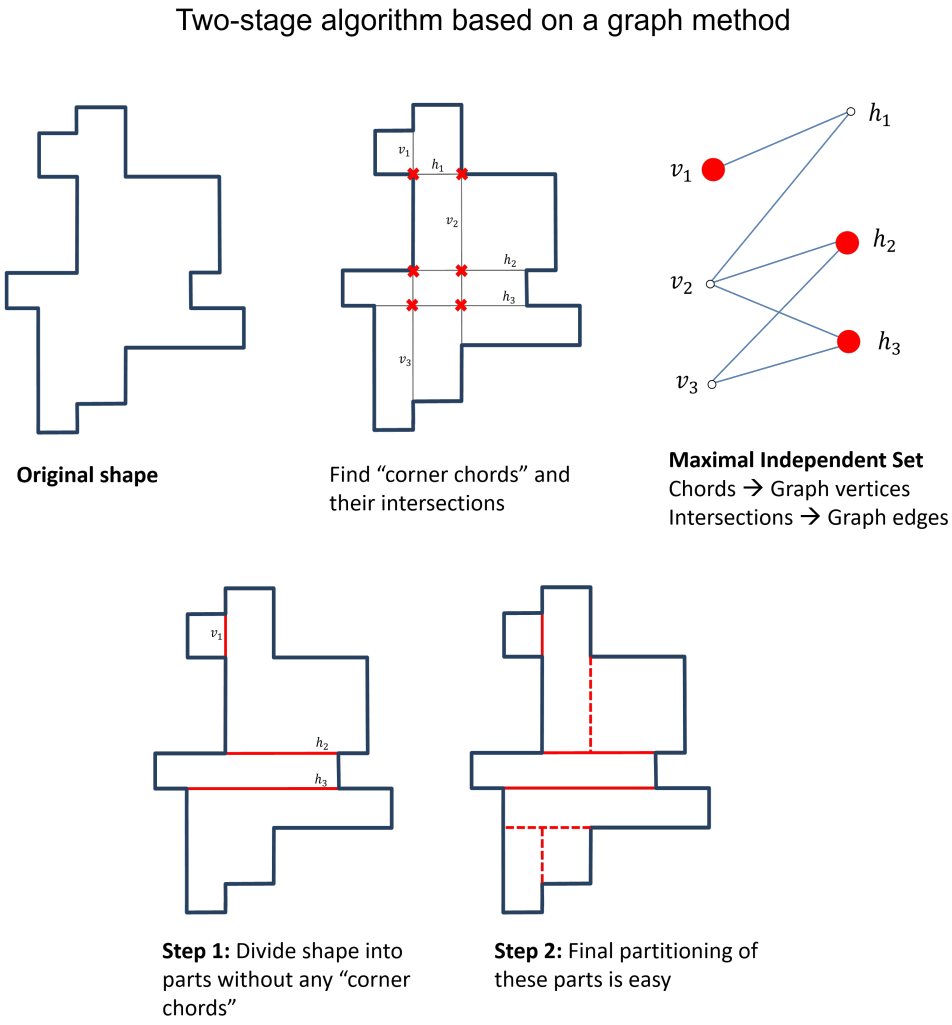


Figure 7: Steps of the optimal decomposition algorithm in 2D. First, we find concave corners and all rectilinear chords that connect them. Then, we find their mutual intersections and in the corresponding graph we find maximal independent set (i.e. the maximal set of non-crossing chords). Once we divide the object along the chords found in step 1, then the final partitioning in step 2 is straightforward.

6.4. Main contribution of the author

- Implementation of the decomposition algorithms
- Design and implementation of a novel algorithm for border reconstruction from decomposed image

7. Close-to-optimal algorithm for rectangular decomposition of 3D shapes

7.1. Citations

- C. Höschl IV and J. Flusser, "Close-to-optimal algorithm for rectangular decomposition of 3d shapes." Submitted to Discrete & Computational Geometry in 2017

Shorter version appeared at

- C. Höschl and J. Flusser, "Decomposition of 3d binary objects into rectangular blocks," in *International Conference on Digital Image Computing: Techniques and Applications DICTA 2016*, (Gold Coast, Australia), pp. 1–8, Nov 2016

7.2. Abstract

In this paper we propose a novel algorithm for the decomposition of 3D binary shapes to rectangular blocks. The aim is to minimize the number of blocks. The theoretically optimal brute-force algorithm is known to be NP-hard and practically unfeasible [38]. We introduce its polynomial suboptimal approximation, which transforms the decomposition problem onto a graph-theoretical problem. We compare its performance with the state of the art Octree and Delta methods. We show by extensive experiments that the proposed method outperforms the existing ones in terms of the number of blocks on a statistically significant level. We also discuss potential applications of the method in image processing.

7.3. Main contribution of the paper

In this paper, we presented an original method of block-wise decomposition in 3D. The proposed method is a polynomial approximative heuristic of the optimal algorithm, which is an NP-complete problem [38]. See Fig. 8 for some examples of decomposed 3D objects.

We proved by large-scale experiments that the proposed method is statistically better than the *3D Generalized Delta Method*, which is the best one among the existing methods in terms of the number of blocks. This determines that potential applications of the proposed method can be found namely in those tasks where it is more desirable to keep the number of blocks as low as possible rather than to minimize the decomposition runtime. See Fig. 9 for a result of the experiment.

Although in the paper we have been dealing with binary shapes only, all methods can theoretically be used for graylevel and color 3D images as well. Like in the 2D counterpart, a rectangular decomposition of 3D objects helps speed-up convolution and integral transformation calculations and can be used in compression as well.

We have created an online tool that visualizes the most important decomposition algorithms and shows the proposed method step-by-step. It is possible to create or upload a custom 3D model and decompose it according to a chosen technique.

7.4. Main contribution of the author

- Design of the proposed novel algorithm
- Running tests
- Implementation of the algorithms
- Development of the online tool for the decomposition

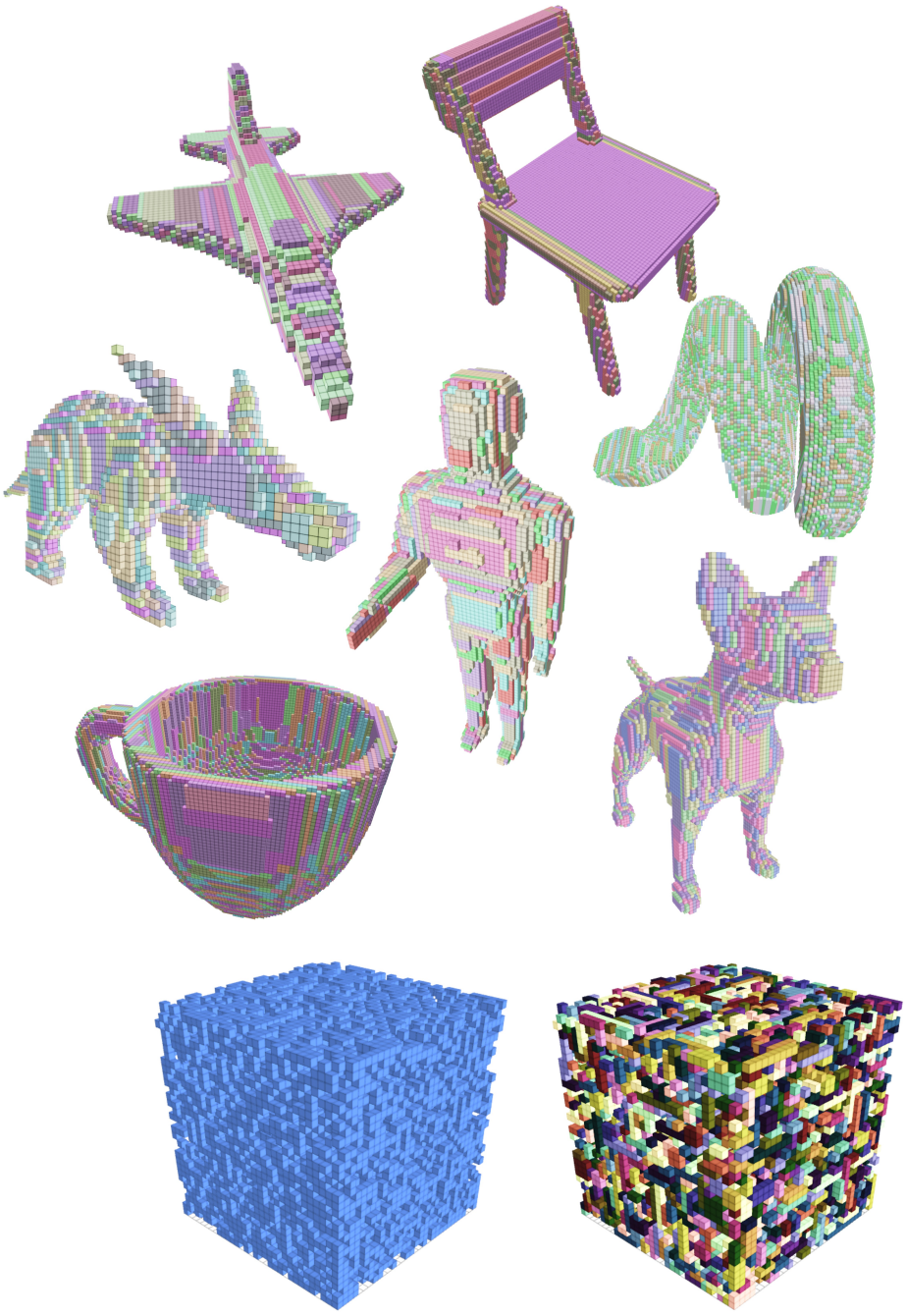


Figure 8: Example of the rectangular decomposition of 3D objects. Above are a few examples of almost 500 objects from the McGill 3D Shape Benchmark database [39]. In the bottom is an instance of a 50% density random cube.

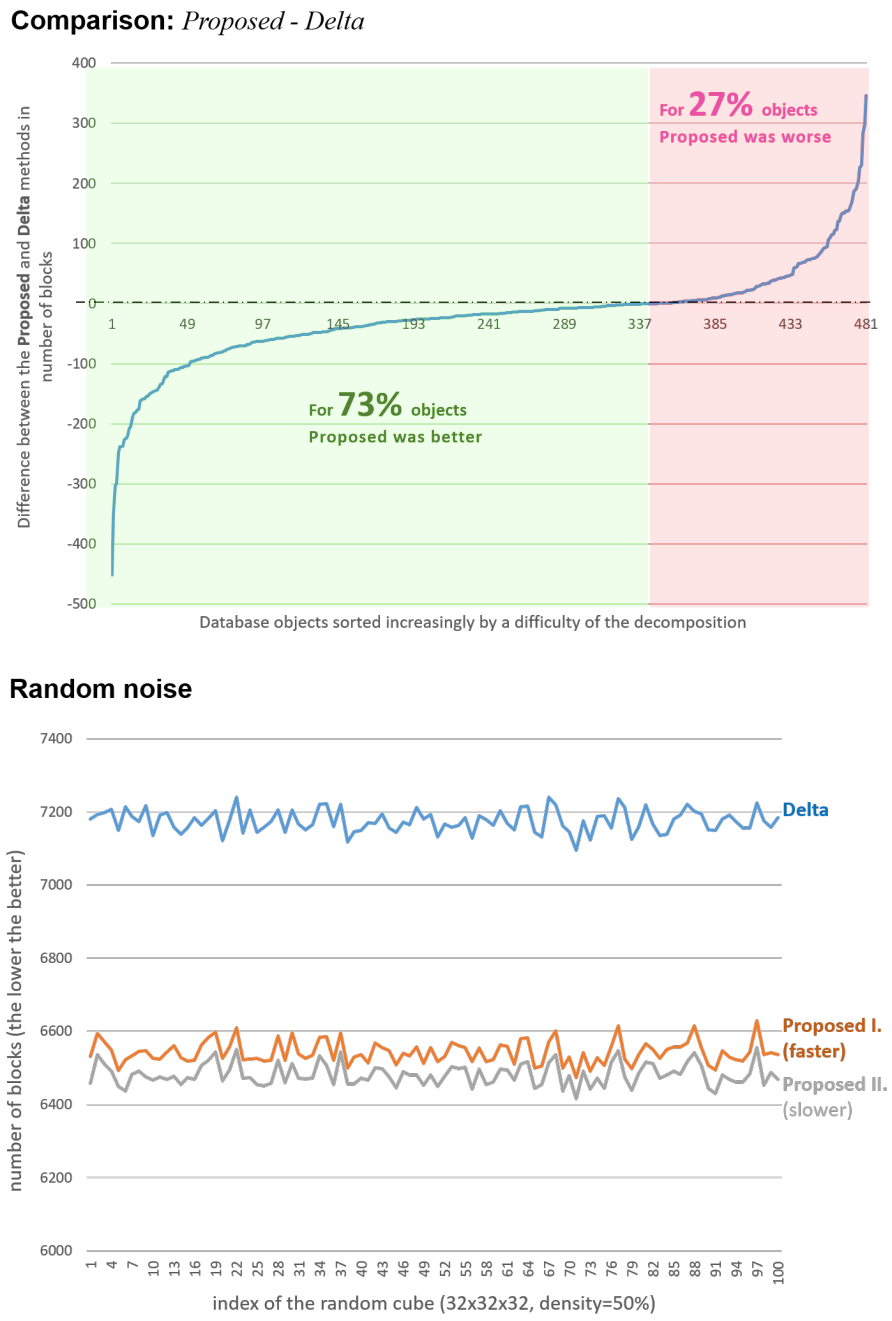


Figure 9: Result of the experiments. The upper chart shows that the proposed method performed better (it has decomposed the objects into fewer blocks) in 73% of 481 cases in the Benchmark database. The bottom chart shows the results of the decomposition of 100 random cubes (see the cube picture in Fig. 8). For the random cubes the proposed methods outperformed the others in all cases (in the chart we show the closest Delta method only).

8. Recognition of Images Degraded by Gaussian Blur

8.1. Citations

- J. Flusser, S. Farokhi, C. Höschl, T. Suk, B. Zitová, and M. Pedone, "Recognition of images degraded by Gaussian blur," *IEEE Transactions on Image Processing*, vol. 25, pp. 790–806, Feb 2016

Shorter version also appeared at

- J. Flusser, T. Suk, S. Farokhi, and C. Höschl IV, "Recognition of images degraded by Gaussian blur," in *Computer Analysis of Images and Patterns CAIP'15* (G. Azzopardi and N. Petkov, eds.), vol. 9256–9257 of *Lecture Notes in Computer Science*, (Valletta, Malta), pp. 88–99, part I, Springer, September 2015

8.2. Abstract

In this paper, we propose a new theory of invariants to Gaussian blur. We introduce a notion of a primordial image as a canonical form of all Gaussian blur-equivalent images. The primordial image is defined in the spectral domain by means of projection operators. We prove that the moments of the primordial image are invariant to Gaussian blur and we derive recursive formulas for their direct computation without actually constructing the primordial image itself. We show how to extend their invariance also to image rotation. The application of these invariants is in blur-invariant image comparison and recognition. In the experimental part, we perform an exhaustive comparison with two main competitors: 1) the Zhang distance and 2) the local phase quantization.

8.3. Main contribution of the paper

We proposed new invariants w.r.t. Gaussian blur, both in frequency and image domains. We showed the performance of the new method in object

recognition and in matching of blurred and noisy templates. Compared to Zhang's method [42], which has been the only Gaussian-blur invariant metric so far, the proposed method is significantly faster and more robust to additive noise while its recognition rate in noise-free cases is fully comparable to Zhang's distance. An additional benefit of the new method is that it can easily be made invariant to translation, rotation, scale, and contrast of the image, which is crucial in many applications and which is not the case of Zhang's method. Last but not least, our method also handles an anisotropic Gaussian blur and is even able to compare images of different sizes.

8.4. Main contribution of the author

- Implementation of the proposed invariants
- Help with the design of experiments

9. Robust histogram-based image retrieval

9.1. Citations

- C. Höschl IV and J. Flusser, "Robust histogram-based image retrieval," *Pattern Recognition Letters*, vol. 69, pp. 72 – 81, 2016

Shorter version also appeared at

- C. Höschl IV and J. Flusser, "Noise-resistant image retrieval," in *22nd International Conference on Pattern Recognition ICPR'14*, (Stockholm, Sweeden), pp. 2972–2977, IEEE, 2014

9.2. Abstract

We present a histogram-based image retrieval method which is designed specifically for noisy query images. The images are retrieved according to histogram similarity. To reach high robustness to noise, the histograms are

Figure 10: Illustration of an image retrieval. The database is searched for the query image affected by a noise.

described by newly proposed features, which are insensitive to Gaussian additive noise in the original images. The advantage of the new method is proved theoretically and demonstrated experimentally on real data.

9.3. Main contribution of the paper

Histograms of a noisy image, both visual appearance and common numerical characteristics, are significantly affected by additive noise in the image. Assuming the noise is independent on the image, the histogram of a noisy image is the clear image histogram convolved with the noise histogram. Provided the noise is Gaussian, we proposed original histogram descriptors, which are invariant w.r.t. the noise. We proved that along with the theoretical invariance the descriptors are sufficiently robust on real images corrupted by thermal and electronic sensor noise. As demonstrated experimentally, the proposed descriptors can be used as features in a histogram-based retrieval if the database and/or query images are heavily noisy and standard descriptors fail (for an illustration see Fig. 10). We confirmed that the retrieval based on the new invariants significantly outperforms the other more traditional methods included in our tests (see Fig. 11 for the results). We have also proved that the method can be used even if the noise distribution is not exactly Gaussian.

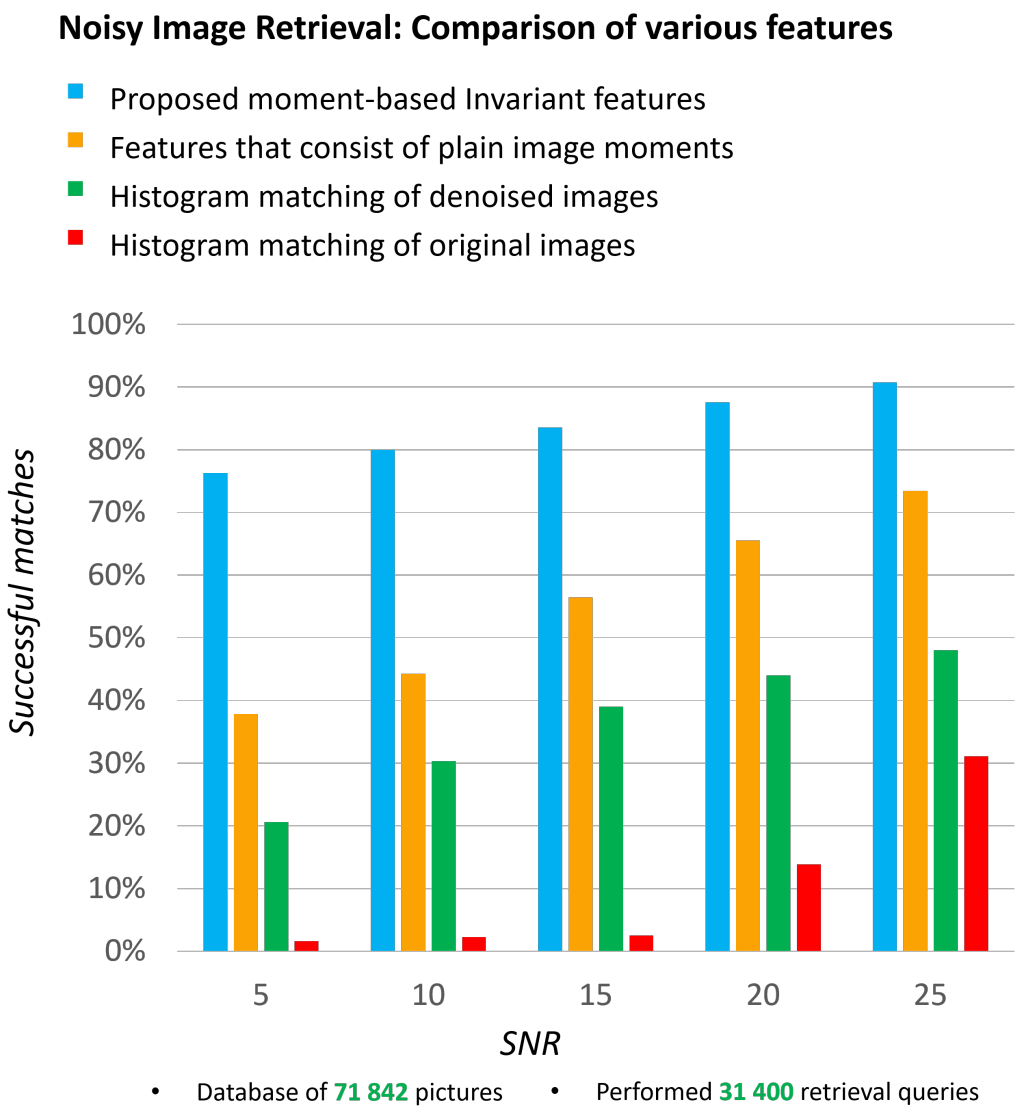


Figure 11: Results of the tests performed on more than 70 000 pictures. The proposed invariant features (blue bars) outperformed other methods for several levels of noise (SNRs).

9.4. Main contribution of the author

- Implementation of the proposed invariants
- Design of the experiments
- Performing tests

10. Main Contribution of the Thesis

The general goal of the Thesis was to contribute to the development of moment-based methods for image analysis. The main contributions of this thesis are the decomposition methods both for 2D and 3D images and a design of new moment-based invariant features.

Contrary to the 2D version where the optimal decomposition algorithm exists and has a polynomial time complexity, in 3D, the optimal algorithm is NP-complete and practically unfeasible. An important contribution of this thesis is a proposition of a new graph-based sub-optimal algorithm that has a polynomial time complexity and yields fewer rectangles than the best heuristics known. We have also created an online tool that does the 3D decomposition of custom models and illustrates the work of the proposed sub-optimal algorithm step-by-step, see [36, 37]. The online tool is available at `http://goo.gl/hAEuCg`, for a screenshot of the tool, see Fig. 13.

The demand of such decomposition algorithms is illustrated in the chart in Fig. 12. It shows the number of downloads of a library "rectangle-decomposition" by Mikola Lysenko[2]. The implementation of the library for the popular runtime environment **Node.js**[3] has been inspired by our work presented in this thesis.

[2]Available online at `https://www.npmjs.com/package/rectangle-decomposition`

[3]See `https://nodejs.org`

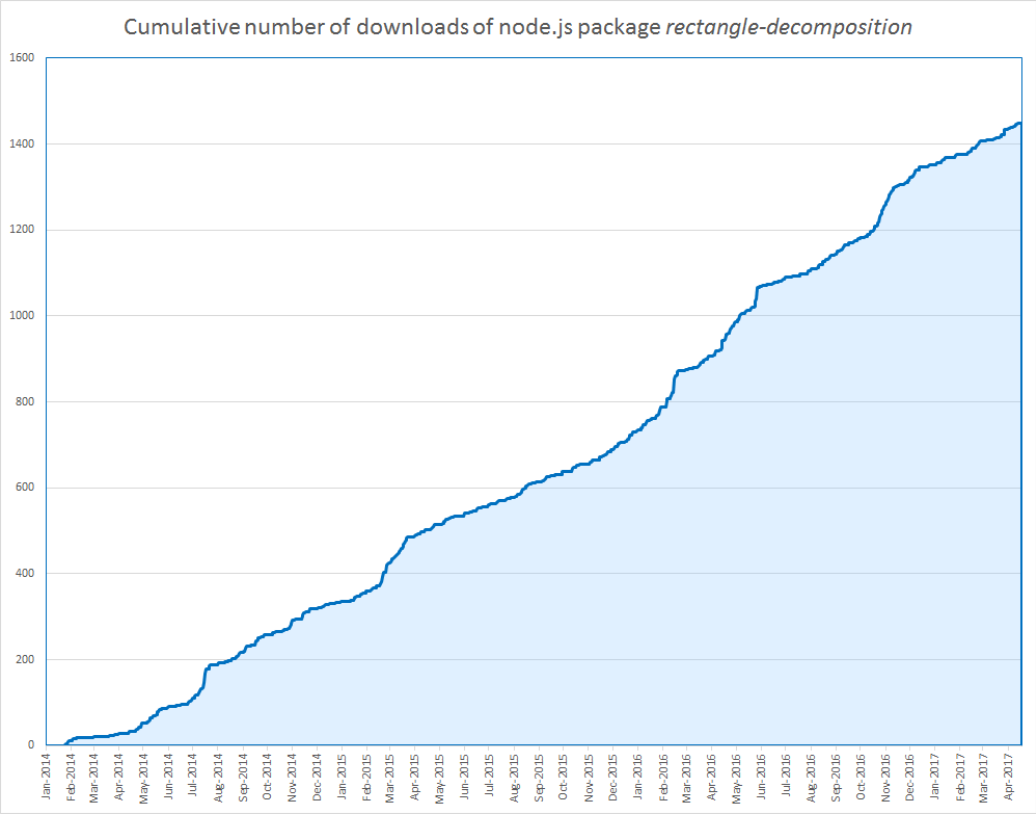


Figure 12: Number of downloads (cumulative) of the Node.js library "rectangular-decomposition" since its release in 2014. The implementation of the library has been based on the work of this thesis [27].

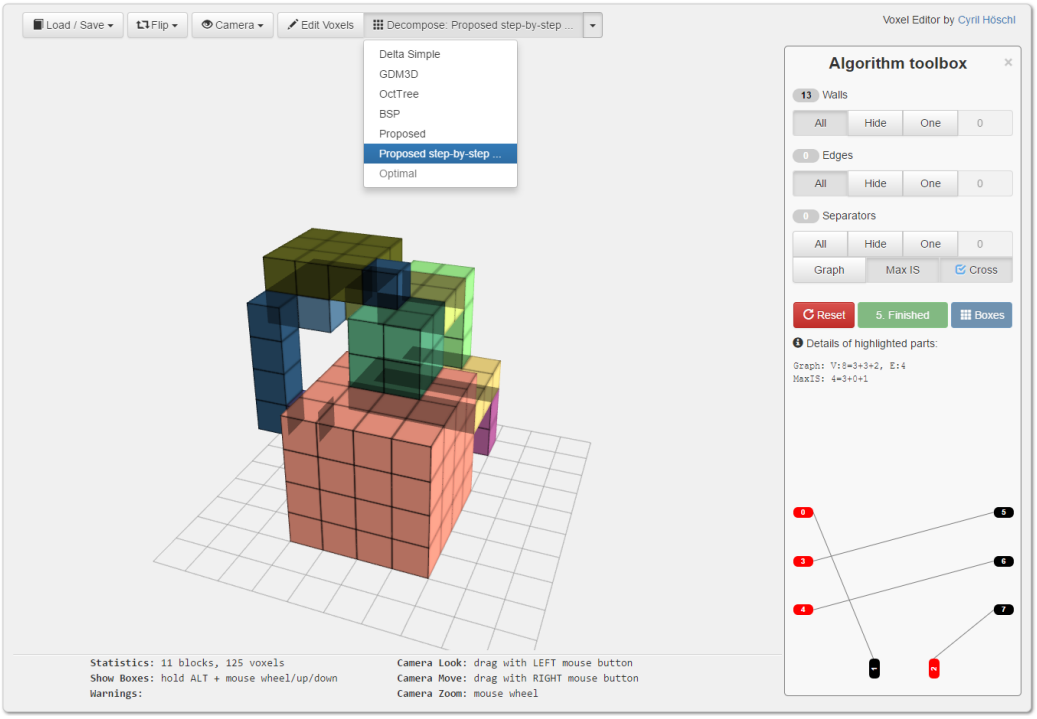


Figure 13: A sample screen shot of our online decomposition tool available at `http://goo.gl/hAEuCg`. The tool performs various decomposition methods and visualizes the proposed algorithm step-by-step.

The second contribution of this thesis is a design of new invariant features based on image moments. We proposed a feature that is invariant to Gaussian blur.

We utilized the principles from the area of blur invariants and derived another feature that is invariant to additive white Gaussian noise. This feature is applied on an image histogram that is affected by the image noise. Our image retrieval experiments on more than 70 thousand pictures show that our proposed invariant method outperforms standard histogram matching and denoising methods convincingly.

References

[1] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*. Academic Press, 2nd ed., 1982.

[2] H. C. Andrews and B. R. Hunt, *Digital Image Restoration*. Englewood Cliffs, New Jersey, USA: Prentice Hall, 1977.

[3] W. K. Pratt, *Digital Image Processing*. New York, USA: Wiley Interscience, 4th ed., 2007.

[4] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*. Prentice Hall, 3rd ed., 2007.

[5] M. Šonka, V. Hlaváč, and R. Boyle, *Image Processing, Analysis and Machine Vision*. Toronto, Canada: Thomson, 3rd ed., 2007.

[6] P. Campisi and K. Egiazarian, *Blind Image Deconvolution: Theory and Applications*. CRC, 2007.

[7] P. Milanfar, *Super-Resolution Imaging*. CRC, 2011.

[8] A. N. Rajagopalan and R. Chellappa, *Motion Deblurring: Algorithms and Systems*. Cambridge University Press, 2014.

[9] A. Rosenfeld, *Picture Processing by Computer*. Academic Press, 1969.

[10] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York, USA: Wiley Interscience, 2nd ed., 2001.

[11] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, 4th ed., 2009.

[12] X. Chen, A. Shrivastava, and A. Gupta, "NEIL: Extracting visual knowledge from web data," in *The 14th International Conference on Computer Vision ICCV'13*, pp. 1409–1416, 2013.

[13] J. Flusser, T. Suk, and B. Zitová, *2D and 3D image analysis by moments*. Chichester, UK Hoboken, NJ: Wiley, 2016.

[14] J. Flusser, T. Suk, and B. Zitová, *Moments and Moment Invariants in Pattern Recognition*. Chichester, U.K.: Wiley, 1rd ed., 2009.

[15] G. A. Papakostas, ed., *Moments and Moment Invariants - Theory and Applications*, vol. 1 of *Gate to Computer Science and Research*. Science Gate Publishing, 2014.

[16] Z. Feng, L. Shang-Qian, W. Da-Bao, and G. Wei, "Aircraft recognition in infrared image using wavelet moment invariants," *Image and Vision Computing*, vol. 27, no. 4, pp. 313–318, 2009.

[17] S. Dominguez, "Image analysis by moment invariants using a set of step-like basis functions," *Pattern Recognition Letters*, vol. 34, no. 16, pp. 2065–2070, 2013.

[18] J. Flusser, T. Suk, and S. Saic, "Recognition of blurred images by the method of moments," *IEEE Transactions on Image Processing*, vol. 5, no. 3, pp. 533–538, 1996.

[19] J. Flusser and T. Suk, "Degraded image analysis: An invariant approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 6, pp. 590–603, 1998.

[20] J. Flusser and B. Zitová, "Combined invariants to linear filtering and rotation," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 13, no. 8, pp. 1123–1136, 1999.

[21] J. Flusser, B. Zitová, and T. Suk, "Invariant-based registration of rotated and blurred images," in *Proceedings of the International Geoscience and Remote Sensing Symposium IGARSS'99*, (Hamburg, Germany), pp. 1262–1264, IEEE, June 1999.

[22] B. Zitová and J. Flusser, "Estimation of camera planar motion from blurred images," in *Proceedings of the International Conference on Image Processing ICIP'02*, (Rochester, New York, USA), pp. 329–332, IEEE, September 2002.

[23] Y. Zhang, C. Wen, Y. Zhang, and Y. C. Soh, "Determination of blur and affine combined invariants by normalization," *Pattern Recognition*, vol. 35, no. 1, pp. 211–221, 2002.

[24] T. Suk and J. Flusser, "Combined blur and affine moment invariants and their use in pattern recognition," *Pattern Recognition*, vol. 36, no. 12, pp. 2895–2907, 2003.

[25] C. Höschl IV and J. Flusser, "Robust histogram-based image retrieval," *Pattern Recognition Letters*, vol. 69, pp. 72 – 81, 2016.

[26] T. Suk, C. Höschl IV, and J. Flusser, "Decomposition of binary images – a survey and comparison," *Pattern Recognition*, vol. 45, no. 12, pp. 4279–4291, 2012.

[27] T. Suk, C. Höschl, and J. Flusser, "Rectangular decomposition of binary images," in *Advanced Concepts for Intelligent Vision Systems: 14th International Conference ACIVS 2012*, (Brno, Czech Republic), pp. 213–224, Springer Berlin Heidelberg, September 2012.

[28] M. F. Zakaria, L. J. Vroomen, P. Zsombor-Murray, and J. M. van Kessel, "Fast algorithm for the computation of moment invariants," *Pattern Recognition*, vol. 20, no. 6, pp. 639–643, 1987.

[29] M. Dai, P. Baylou, and M. Najim, "An efficient algorithm for computation of shape moments from run-length codes or chain codes," *Pattern Recognition*, vol. 25, no. 10, pp. 1119–1128, 1992.

[30] B. C. Li, "A new computation of geometric moments," *Pattern Recognition*, vol. 26, no. 1, pp. 109–113, 1993.

[31] I. M. Spiliotis and B. G. Mertzios, "Real-time computation of two-dimensional moments on binary images using image block representation," *IEEE Transactions on Image Processing*, vol. 7, no. 11, pp. 1609–1615, 1998.

[32] J. Flusser, "Refined moment calculation using image block representation," *IEEE Transactions on Image Processing*, vol. 9, no. 11, pp. 1977–1978, 2000.

[33] C.-H. Wu, S.-J. Horng, and P.-Z. Lee, "A new computation of shape moments via quadtree decomposition," *Pattern Recognition*, vol. 34, no. 7, pp. 1319–1330, 2001.

[34] J. H. Sossa-Azuela, C. Yáñez-Márquez, and J. L. Díaz de León Santiago, "Computing geometric moments using morphological erosions," *Pattern Recognition*, vol. 34, no. 2, pp. 271–276, 2001.

[35] T. Suk and J. Flusser, "Refined morphological methods of moment computation," in *20th International Conference on Pattern Recognition ICPR'10*, (Istanbul, Turkey), pp. 966–970, IEEE, August 2010.

[36] C. Höschl IV and J. Flusser, "Close-to-optimal algorithm for rectangular decomposition of 3d shapes." Submitted to Discrete & Computational Geometry in 2017.

[37] C. Höschl and J. Flusser, "Decomposition of 3d binary objects into rectangular blocks," in *International Conference on Digital Image Computing: Techniques and Applications DICTA 2016*, (Gold Coast, Australia), pp. 1–8, Nov 2016.

[38] V. J. Dielissen and A. Kaldewaij, "Rectangular partition is polynomial in two dimensions but NP-complete in three," *Information Processing Letters*, vol. 38, no. 1, pp. 1–6, 1991.

[39] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson, "Retrieving articulated 3-d models using medial surfaces," *Mach. Vision Appl.*, vol. 19, pp. 261–275, May 2008.

[40] J. Flusser, S. Farokhi, C. Höschl, T. Suk, B. Zitová, and M. Pedone, "Recognition of images degraded by Gaussian blur," *IEEE Transactions on Image Processing*, vol. 25, pp. 790–806, Feb 2016.

[41] J. Flusser, T. Suk, S. Farokhi, and C. Höschl IV, "Recognition of images degraded by Gaussian blur," in *Computer Analysis of Images and Patterns CAIP'15* (G. Azzopardi and N. Petkov, eds.), vol. 9256–9257 of *Lecture Notes in Computer Science*, (Valletta, Malta), pp. 88–99, part I, Springer, September 2015.

[42] Z. Zhang, E. Klassen, and A. Srivastava, "Gaussian blurring-invariant comparison of signals and images," *IEEE Transactions on Image Processing*, vol. 22, no. 8, pp. 3145–3157, 2013.

[43] C. Höschl IV and J. Flusser, "Noise-resistant image retrieval," in *22nd International Conference on Pattern Recognition ICPR'14*, (Stockholm, Sweeden), pp. 2972–2977, IEEE, 2014.

Pattern Recognition 45 (2012) 4279–4291

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Decomposition of binary images—A survey and comparison

Tomáš Suk*, Cyril Höschl IV, Jan Flusser

Institute of Information Theory and Automation of the ASCR, Pod vodárenskou věží 4, 182 08 Praha 8, Czech Republic

ARTICLE INFO

Article history:
Received 10 February 2012
Received in revised form
24 April 2012
Accepted 16 May 2012
Available online 26 May 2012

Keywords:
Binary image decomposition
Distance transform
Quadtree
Bipartite graph
Image compression
Moment computation
Fast convolution

ABSTRACT

We present an overview of the most important methods that decompose an arbitrary binary object into a union of rectangles. We describe a run-length encoding and its generalization, decompositions based on quadtrees, on mathematical morphology, on the distance transform, and a theoretically optimal decomposition based on a maximal matching in bipartite graphs. We compare their performance in image compression, in moment computation and in linear filtering. We show that the choice is always a compromise between the complexity and time/memory consumption. We give advice how to select an appropriate method in particular cases.

1. Introduction

It is intuitively clear and well known that binary images can be represented in a more efficient way than just as a full-sized matrix consisting of zeros and ones. It is also clear that the terms "good representation" and "optimal representation" cannot be generally defined and are always dependent on what we are going to do with the object. Usually there are two basic requirements on the representation—small time/memory consumption and possibility of fast further computation (which is application-dependent). Sometimes we also require fast recovery of the original.

Regardless of the particular purpose, the methods of binary image representation can be divided into two groups referred as *decomposition methods* and *boundary-based methods*. Boundary-based methods employ the property that the boundary of a binary object contains a complete information on the object, all other pixels are redundant. Provided that the boundary consists of much less pixels than the whole object (which applies to "normal" shapes but does not hold true in general), it provides efficient non-redundant representation. Individual boundary-based methods differ from each other by a boundary definition (outer or inner boundary), by a discrete topology used (4-pixel or 8-pixel connectivity) and by the way how the boundary is encoded and stored (chain codes and various piece-wise approximations are mostly used for this purpose).

Decomposition methods try to express the object as a union of simple disjoint subsets called *blocks* or *partitions* which can be effectively stored and consequently used for required processing. Having a binary object B (by a binary object we understand a set of all pixels of a binary image whose values equal one), we decompose it into $K \geq 1$ blocks $B_1, B_2, \ldots, B_K$ such that $B_i \cap B_j = \emptyset$ for any $i \neq j$ and $B = \bigcup_{k=1}^{K} B_k$. Although in a continuous domain we may consider various shapes of the blocks (convex, star-shaped, hexagonal, rectangular, etc., see [1]), all decomposition methods that perform in a discrete domain use only rectilinear rectangular or square blocks because of a native rectangular structure of the discrete image domain. The methods differ from one another namely by the decomposition algorithms.

The power of any decomposition method depends on its ability to decompose the object into a small number of blocks in a reasonable time. Most authors have measured the decomposition quality just by the number of blocks K, while ignoring the complexity of the algorithms (it should be noted that there exist a few other criteria such as the "minimum ink" criterion which minimizes the overall length of the inner boundary but they are out of the scope of this paper). There is a common belief that such decomposition that minimizes K is the optimal one. This criterion is justified by the fact that the complexity of subsequent calculations uses to be $\mathcal{O}(K)$ and compression ratio (if the decomposition is used for compression purposes) also increases as the number of blocks decreases. However, this viewpoint may be misleading. Simple algorithms produce relatively high number of blocks but

* Corresponding author. Tel.: +420 26605 2231; fax: +420 28468 0730.
E-mail addresses: suk@utia.cas.cz (T. Suk), hoschl@utia.cas.cz (C. Höschl IV), flusser@utia.cas.cz (J. Flusser).

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.05.012

perform fast, while more sophisticated decomposition methods end up with small number of blocks but require more time. Even if the decomposition is performed only once per object in most tasks and can be done off-line, the time needed for decomposing the image is often so long that it substantially influences the efficiency of the whole method.

Image rectangular decomposition has found numerous straightforward applications in image compression methods and formats (RLE, TIFF, BMP and others) and in calculation of image features (mainly moments and moment invariants) used subsequently for object description and recognition [2–9]. Other applications may be found in image spatial filtering and restoration, in integrated circuits design and in other areas. Convolution with a constant rectangular mask can be performed in $\mathcal{O}(1)$ time per pixel if the matrix of partial sums is precomputed. If the mask is constant or piecewise constant but not rectangular, its support can be decomposed into rectangles and the convolution in one pixel can be calculated in $\mathcal{O}(K)$ time as a sum of partial $\mathcal{O}(1)$ convolutions. In VLSI design, the decomposition problem appeared many years ago—the masks are rectilinear polygons (often very complex) which should be decomposed into rectangles in such a way that the pattern generator can effectively generate the mask. The time needed for a mask generation is proportional to the number of rectangles, so it is highly desirable to minimize their number [10,11].

The aim of this paper is to present a survey and a comparison of existing decomposition methods. To ensure an unbiased comparison, all reviewed methods were implemented on the same platform and run on the same computer. We compare their performance in three common tasks—loss-less compression, calculation of image moments and convolution with a binary mask. We show that there is no "generally optimal" decomposition method and explain the pros and cons of individual algorithms.

2. Decomposition methods

In this section, we present a brief survey of the most common decomposition methods. Their performance in real-data experiments is compared in Sections 3–5.

2.1. Decomposition into row segments

Decomposition of an object into rows or columns is the most straightforward and the oldest method. The blocks are continuous row segments for which only the coordinate of the beginning and the length is stored. In image compression, this has been known as run-length encoding (RLE). This principle and its modifications (CCITT, PackBits) are used in several image formats such as TIFF and BMP. In feature calculation, Zakaria et al. [2] used the same representation for fast computation of image moments of convex shapes and called it "Delta-Method" (DM) since the lengths of the row segments were labeled by the Greek letter δ. The method was slightly improved by Dai et al. [3] and generalized for non-convex shapes by Li [4].

The decomposition into rows is very fast but leads to the number of blocks which uses to be (much) higher than the minimal decomposition. A simple but powerful improvement of the Delta-Method was proposed by Spiliotis and Mertzios [5] and adopted by Flusser [6]. This "Generalized Delta-Method" (GDM) employs a rectangular-wise object representation instead of the row-wise one. The adjacent rows are compared and if there are some segments with the same beginning and end, they are unified into a rectangle (see Fig. 1). For each rectangle, the coordinates of its upper-left corner, the length and the width are stored. GDM is only slightly slower than DM while producing (sometimes significantly) less number of blocks. Surprisingly, under our knowledge this method has not been implemented in any commercial image format.

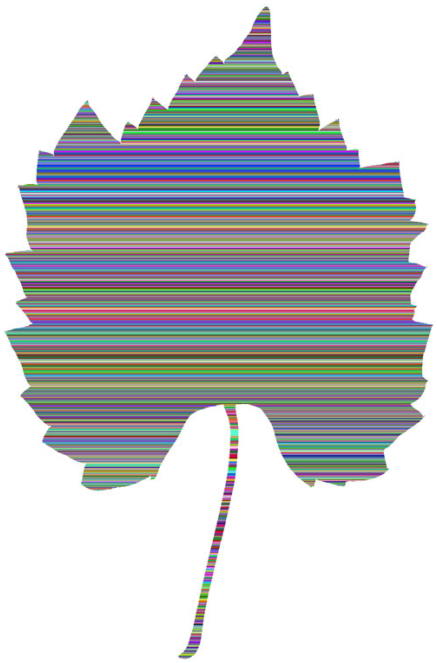


Fig. 1. Decomposition of the leaf image by the Generalized Delta-Method (GDM). The size of the original image is 1196×1828 pixels. The adjacent rows of the same length are unified into blocks (1963 blocks in total).

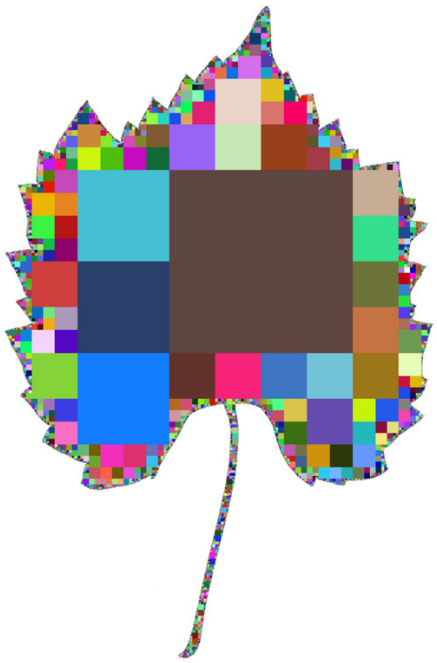


Fig. 2. Decomposition of the leaf image by the quadtree method (8911 blocks in total).

2.2. Quadtree decomposition

Quadtree decomposition (QTD) is a popular hierarchical decomposition scheme used in several image processing areas including representation and compression [12], spatial transformations [13] and feature calculation [7]. In its basic version, QTD works with square images of a size of a power of two. If this is not the case, the image is zero-padded to the nearest such size. The image is iteratively divided into four quadrants. Homogeneity of each quadrant is checked and if the whole quadrant lies either in the object or in the background it is not further divided. If it contains both object and background pixels, it is divided into quadrants and the process is repeated until all blocks are homogeneous. The decomposition can be efficiently encoded into three-symbol string, where 2 means division, 1 means a part of the object and 0 stands for the background. An example of the quadtree decomposition is in Fig. 2.

The algorithm always yields square blocks which may be advantageous for some purpose but usually leads to a higher number of blocks than necessary. It would be possible to implement a backtracking and to unify the adjacent blocks of the same size into a rectangle but this would increase the complexity. Since the speed is the main advantage of this method, the backtracking is mostly not employed here. A drawback of this decomposition algorithm is that the division scheme is not adapted with respect to the content of the image but it is defined by absolute spatial coordinates. Hence, the decomposition is not translation-invariant and may lead to absurd results when for instance a large single square is uselessly decomposed up to individual pixels. We may use a bintree or other trees producing non-square blocks but it does not overcome this principal weakness.

2.3. Morphological decomposition

In order to better adapt the decomposition to the image content, Sossa-Azuela et al. [8] published a decomposition algorithm based on a *morphological erosion*. The erosion is an operation, where a small structural element (here 3×3 square is used) moves over the image and when the whole element lies in the object, then the central pixel of the window is preserved in the object, otherwise it is assigned to the background. So, each erosion shrinks the object by one-pixel boundary layer. The decomposition works in an iterative manner: it finds the largest square inscribed in the object, removes it and looks for the largest square inscribed in the rest of the object. This outer loop is repeated until the object is completely decomposed. An intermediate object decomposition after two outer loops can be seen in Fig. 3, the final decomposition is in Fig. 4.

Although the original method [8] considers decomposition into squares only, it can be generalized also to rectangles which decreases the number of the resulting blocks. The inner loop serves for finding the center and the size of the largest inscribed square/rectangle. We repeat the erosion until the whole object disappears and count the number of erosions s. Then a $(2s-1) \times (2s-1)$ square can be inscribed into the object and it forms one block of the decomposition. The pixels of the object before the last disappearing erosion are potential centers of the inscribed square. Theoretically, we can choose one of them randomly, but the "corner" pixels provide better odds to more compact rest of the object. If the potential square centers create a line segment, then the corresponding inscribed squares can be unified into a rectangle.

It is possible, especially in the last steps of the method, that several of the identically sized squares (overlapping as well as non-overlapping) can be inscribed into different places of the object. Of course, it is possible to inscribe and remove one of them, repeat the erosions, inscribe and remove another one, etc.

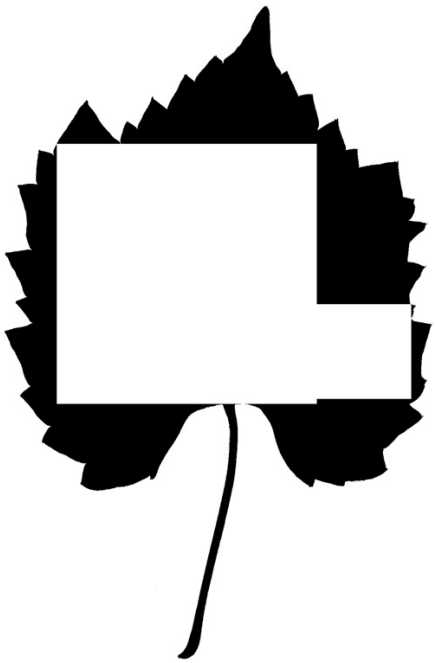


Fig. 3. The leaf image after removing two largest square blocks.

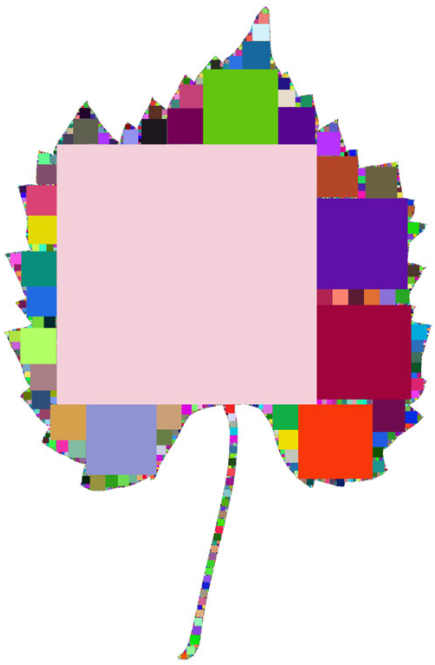


Fig. 4. Morphological decomposition of the leaf image to squares (7489 blocks in total).

A better approach is, after inscribing and removing one of them, to remove the centers of the squares that would overlap this one and search a center of another block of this size without repeating the erosions.

Vizireanu (e.g. [14]) generalized the morphological decomposition for other applications as skeleton computation or image interpolation. The skeleton can be used for image compression,

but its utilization for feature computation is difficult. The interpolation of an image between two frames in a video sequence can be computed even for gray-scale images.

2.4. Distance transform decomposition

If the object is sufficiently compact, i.e. the largest inscribed square is bigger than a certain minimum size, we can speed up the previous algorithm by means of the *distance transform* (DT), which we use for the finding of the centers of the inscribed rectangles [9]. In morphological decomposition, we must repeat the erosions s-times for finding $(2s-1)\times(2s-1)$ inscribed square, while the distance transformation with a suitable metric can be calculated only once. DT of a binary image is an image, where each object pixel shows the distance to the nearest boundary pixel and the background pixels are zero [15].

DT strongly depends on the metric used for the distance measurement. We use a simplified version of the Seaidoun's algorithm [16] for the chessboard metric

$$d(a,b)=\max\{|a_x-b_x|,|a_y-b_y|\}. \quad (1)$$

We successively search the image from the left, right, top and bottom, count distances from the last boundary pixel and calculate the minimum from the four directions. The result is DT, the maximum of the result equals s for the inscribed square $(2s-1)\times(2s-1)$ and the pixels with this maximum value are potential centers of the inscribed squares.

We use an improved version of DT inspired by [17]. If we change only a small part of the original image, then upgrading its close neighborhood is sufficient. In our case, if we remove a rectangle from the image, then the rectangle is zeroed and DT is recomputed in a small frame around it. If the frame in some distance d from the rectangle is not changed, then the rest of DT is left unchanged. In Fig. 5, we can see the visualization of DT of the leaf image.

Similarly to the morphological decomposition, the algorithm consists on iterative repeating the following loop until the object is fully decomposed. In each run, the largest inscribed rectangular block is found. Its potential center(s) are the pixels with the maximum values of DT. If the maximum s is unique, i.e. there is a single pixel with value s only, then a square $(2s-1)\times(2s-1)$ is inscribed. This is however a rare case, often the maximum is not unique and the choice of the block center is ambiguous. We try to keep the blocks as large as possible. Hence, if there is a 2×2 square of the maxima, an even-sized square $2s\times 2s$ can be inscribed into the object. If the potential square centers create a line segment (with a single or double-pixel width), then the corresponding squares are unified into one rectangle like in the morphological method. At the end of the loop, the inscribed rectangle is removed from the object and the procedure starts the next loop, which is applied to the rest of the image. The decomposed leaf image is shown in Fig. 6.

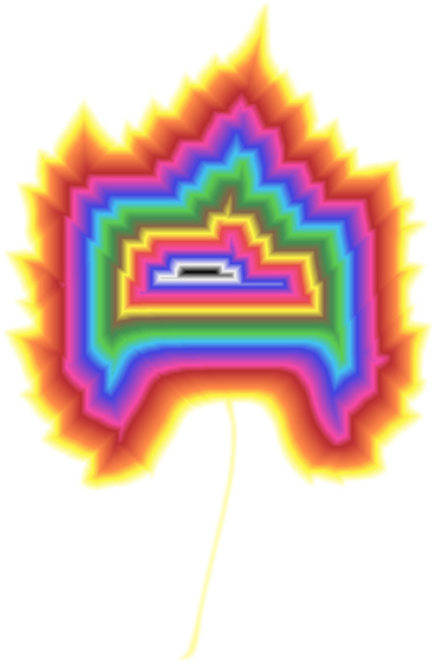


Fig. 5. The distance transform of the leaf image. The pixels are labeled by pseudocolors according to the DT values.

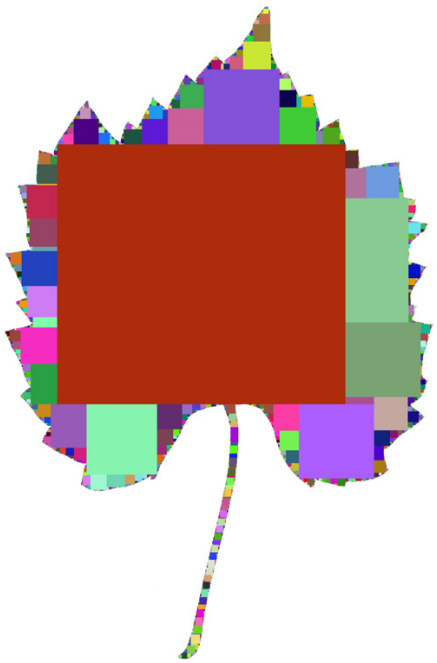


Fig. 6. The distance transform decomposition of the leaf image (2482 blocks in total).

Both morphological and DT decompositions end up with the same set of blocks. However, they are still only sub-optimal even on many simple shapes. This is because a sequence of locally optimal steps that these "greedy" algorithms apply to an image (placing always the largest inscribed rectangle) may not yield an optimal solution. As soon as a block is created, it cannot be removed any more, because the methods do not include any backtracking. These two algorithms differ from each other by computing complexity. Erosion is a relatively complex operation, while the DT can be calculated faster thanks to its simple upgrading. Anyway, both methods perform slower comparing to the previous decomposition algorithms.

2.5. Graph-based decomposition

A large group of decomposition algorithms appeared in 1980s in computational geometry [1]. Surprisingly, they have not received almost any attention from image analysis community. Their formulation was usually much more general than ours. They tried to decompose general polygons into specific polygonal components (convex polygons, star-shape polygons, triangles,

generally oriented rectangles, etc.). A common feature of these methods was that they transformed the decomposition problem to a graph partitioning problem and employed tools known from graph theory. The only subgroup relevant to our purposes is a decomposition of a digital polygon into rectilinear rectangles. An algorithm which was proved to be optimal in terms of the number of blocks was independently proposed in the same form by three different authors [18–20] (in this order) and later discussed by [21,22] and others. The method (denoted here as FER) works for any object even if it contains holes. As will be discussed later, individual versions of the algorithm differ from one another only by the implementation of one step, which may influence the complexity but not the total number of the blocks.

The method performs hierarchically on two levels. On the first level, we detect all "concave" vertices (i.e. those having the inner angle 270°) of the input object and identify pairs of "cogrid" concave vertices (i.e. those having the same horizontal or vertical coordinates). Then we divide the object into subpolygons by constructing chords which connect certain cogrid concave vertices. It is proved in [20] and other papers that the optimal choice of the chord set is such that the chords do not intersect each other and their number is maximum possible.

The problem of optimal selection of the chords is equivalent to the problem of finding the maximal set of independent nodes in a graph, where each node corresponds to a chord and two nodes are connected with an edge, if the two chords have a common point (either a vertex or an intersection). Generally, this problem is NP-complete, but our graph is a bipartite one, since any two horizontal (vertical) edges cannot intersect one another. In a bipartite graph, this task can be efficiently resolved. First, we find a maximal matching, which is a classical problem in graph theory, whose algorithmic solution in a polynomial time has been published in various versions. Some of them are optimized with respect to the number of edges, the others with respect to the number of vertices (see [23–25,21] for some examples of particular algorithms). For binary object decomposition, it is impossible to choose one that would be time-optimal for any object, because the number of vertices and/or edges of the graph depends on the shape of the object. We implemented the algorithm by Edmonds and Karp [25] which is based on the Maximum Network Flow and is linear w.r.t. the number of vertices and quadratic w.r.t. the number of edges.

As soon as the maximal matching has been constructed, the maximal set of independent nodes can be found much faster than the maximal matching itself—roughly speaking, the maximal independent set contains one node of each matching pair plus all isolated nodes plus some other nodes, which are not included in the matching but still independent. As a result, we obtain a set of nodes that is unique in terms of the number but ambiguous in terms of particular nodes. However, this ambiguity does not play any role—although each set leads to different partitioning, the number of partitions is always the same. Hence, at the end of the first level, the object is decomposed into subpolygons, which do not contain any cogrid concave vertices (see Fig. 7).

The second level is very simple. Each subpolygon coming from the first level is further divided. From each its concave vertex, a single chord is constructed such that this chord terminates either on the boundary of the subpolygon or on the chord constructed earlier. This is a sequential process in which each concave vertex is visited only once. We may choose randomly between two possible chords offered in each concave vertex. After that, the subpolygon is divided into rectangles, because rectangle is the only polygon having no concave vertices (see Fig. 8).

The strength of this algorithm is in the fact that it guarantees minimizing the number of decomposing rectangles regardless of the particular choices on the both levels. On the other hand, one may expect slower performance than in the previous algorithms namely because of expensive finding the maximum set of independent graph nodes. Since the optimal partitioning is not unique on both levels, one could require additional constraints, such as

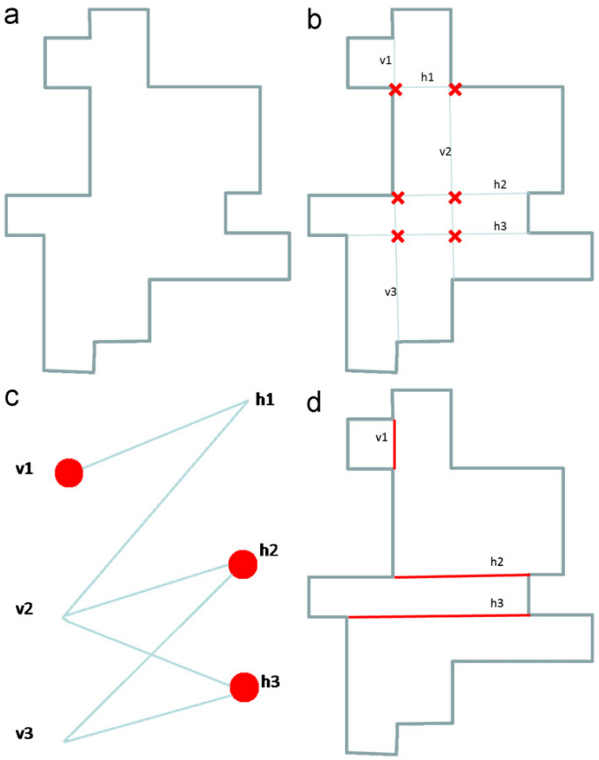


Fig. 7. (b) First-level decomposition of the object (a) by chords connecting the cogrid concave vertices. The crosses indicate the chord intersections. (c) The corresponding bipartite graph with a maximum independent set of three vertices and (d) the object decomposition.

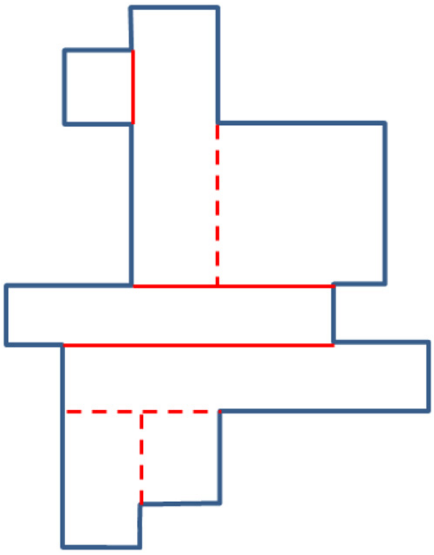


Fig. 8. Second-level decomposition of a subpolygon. From each concave vertex a single chord of arbitrary direction is constructed.

minimum length of inner boundary, but this would lead to further increasing of the complexity.

If the object does not have any cogrid concave vertices or if the maximum independent set of nodes (or, more precisely, at least one of such sets) contains only the nodes corresponding to the horizontal (or vertical) chords, then also all chords in the second level may be constructed in the same direction and, consequently, FER decomposition leads exactly to the same partitioning as GDM applied in that direction (see Fig. 9 for an example). This helps us not only to understand when GDM is strong, but also to interpret the idea behind FER algorithm—it may be viewed as a "local GDM" properly switching between the directions. An example of FER decomposition is in Fig. 10.

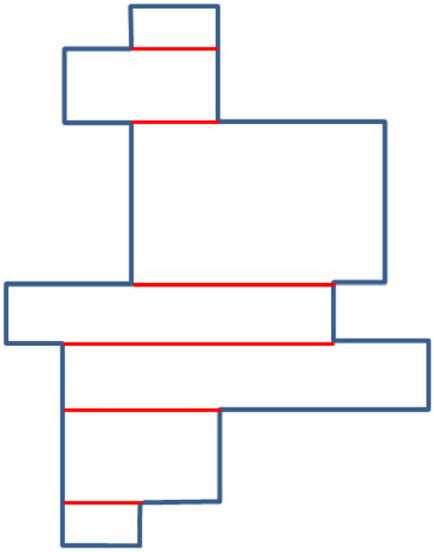


Fig. 9. An example where both FER and GDM yield the same decompositions.

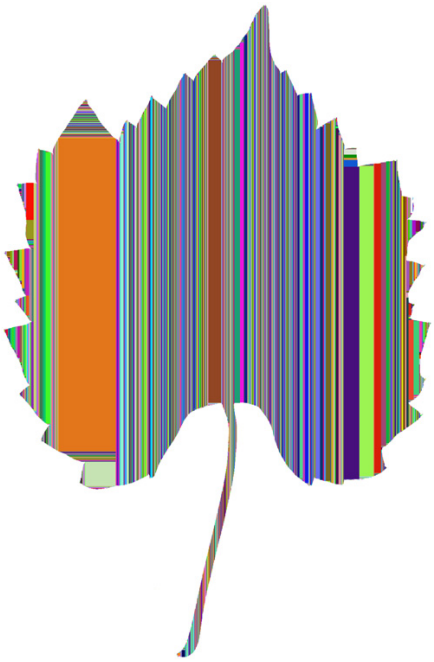


Fig. 10. FER decomposition of the leaf image (1748 blocks in total).

3. Experimental comparison—image compression

In this section, we compare the performance of the above decomposition methods in binary image compression. The experiments were performed on the publicly available LEAF database [26], which is a database of 795 scanned and binarized leaves of trees and shrubs of vegetation growing in the Czech Republic (see Fig. 11).[1] All methods were implemented in C++ language and run on a PC with Intel Core 2 Duo, 2.8 GHz CPU and Windows 7 Professional.

3.1. Efficiency of the decomposition

First we monitor and compare two parameters: the number of blocks and the corresponding decomposition time for GDM, QTD, DT and FER methods, respectively. The data presented in Table 1 are cumulative for all objects in the database. The time was always measured just of the decomposition itself, no input/output operations were included. The minimum number of blocks was achieved by FER, as one expects from the theory. This is of course on the expense of the time, but surprisingly the time is lower than that of DT and only five times higher than the time of QTD. The winner of this test is GDM method yielding only a slightly worse number of blocks than FER but in the by far shortest time. On the other hand, QTD produces the highest block number and the DT is the slowest.

It is interesting that the differences among these decomposition methods are observable even on very small and simple images. For example, the 8×8 object in Fig. 12a was decomposed by (DM, GDM, QTD, DT, and FER) into (10, 5, 23, 6, and 4) rectangular blocks, respectively.

3.2. Compression ratio

Although the decomposition itself makes a substantial compression, we further increase the compression ratios of all methods by a proper block ordering. We propose a new file format denoted as BLK. It uses three types of compression: the blocks from DT are grouped according their size, this allows to store the size only once per each group and then to store just upper left corner of each block. The long narrow blocks from GDM and FER are sorted by the coordinates in the "narrow" direction and sizes and coordinate differences in this direction are encoded in the decreased number of bits. QTD uses its traditional three-symbol encoding. The label of the actual decomposition method is stored in the file header along with other auxiliary parameters.

We compared compression ratios of BLK format (with all these four decomposition methods) to commercial formats. As we already explained, it does not reflect only the number of blocks, since other factors playing role there. We calculated average compression ratios (ACR) over the LEAF database. ACR is a ratio of the size of all files in the database in the specific format and the size without any compression. The results are in Table 2. In this experiment, it is not meaningful to measure the time, because it inherently includes I/O operations. Since we do not have an access to the source codes of commercial compressions, it would be impossible to ensure an objective time comparison.

[1] The database exists in two versions—original (the leaves with petioles, high resolution) and simplified (the leaves without petioles, downsampled by a factor 2). Table 1 refers to the simplified version, Table 2 to the original one.

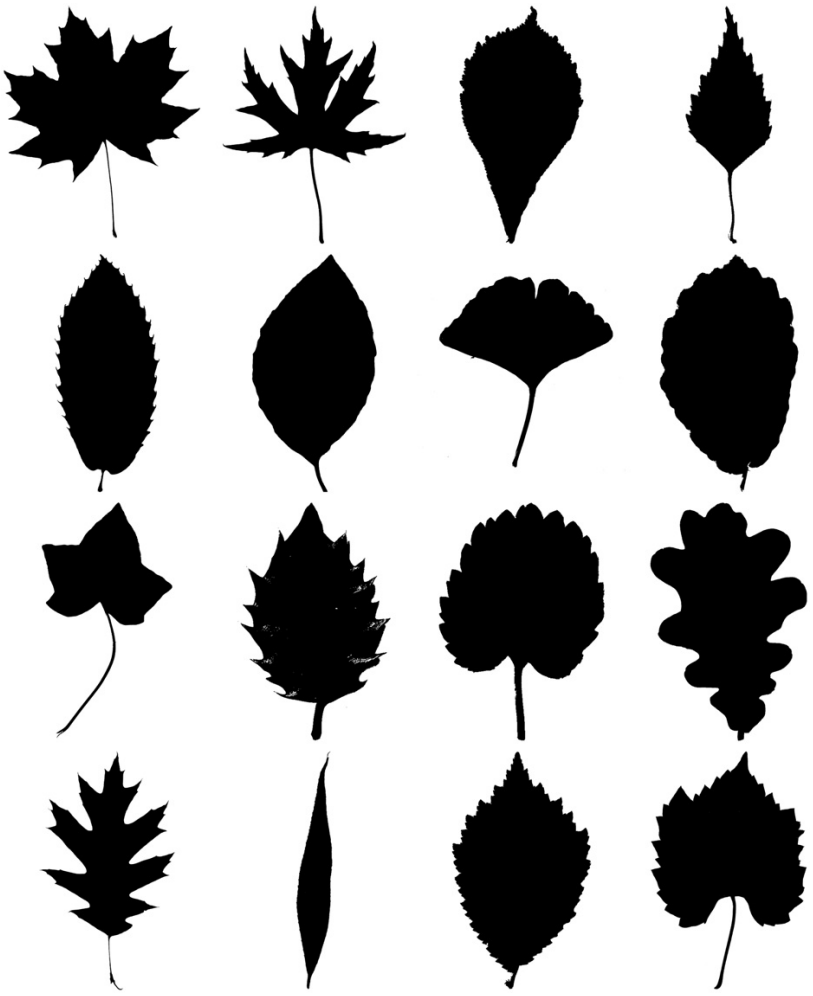


Fig. 11. Examples of the LEAF database (*Acer platanoides*, *Acer saccharinum*, *Aesculus hippocastanum*, *Betula pendula*, *Castanea sativa*, *Fagus sylvatica*, *Ginkgo biloba*, *Hamamelis virginiana*, *Hedera helix*, *Ilex aquifolium*, *Morus alba*, *Quercus robur*, *Quercus rubra*, *Salix alba*, *Ulmus glabra*, and *Vitis riparia*).

Table 1
The number of blocks and decomposition time achieved on the LEAF database.

Method	# blocks	# blocks (%)	Time (s)
GDM	419,489	112	1.3
QTD	1,913,275	511	7.2
DT	545,528	146	50.3
FER	374,149	100	37.5

The best performance of QTD was achieved namely because its three-symbol encoding in the BLK format is very efficient. This yields good compression even if the number of blocks is high. The ACR of FER and GDM is slightly worse, but still very good; FER method is better because it guarantees the minimum number of blocks. The result of DT is still good comparing to TIFF and was achieved by efficient encoding of one-pixel blocks. However, this method is very time-consuming. The only commercial format providing a comparable ACR is PNG with a deflate method, others provide worse compression ratios than all BLK methods.

Based on these two measurements a general recommendation can be as follows: QTD method yields good compression ratio because of efficient implementation in BLK format, but we should keep in mind that the number of blocks was four times higher and the decomposition time even six times higher than GDM offering a very good compromise between compression ratio and decomposition speed. FER and DT methods are suitable mainly for applications, where the compression is performed only once (usually off-line) and thus the time of the compression is not critical. FER provides minimum number of blocks and slightly better ACR and time of compression.

The results of the previous experiments are statistically significant and can be generalized. However, for specific shapes the results may be different. The same experiment was carried out on special images “Transmitter” and “Labyrinth” to illustrate the behavior of the algorithms in extreme cases. Transmitter is decomposed into very high number of blocks (7207 in an optimal case), while Labyrinth represents an opposite case with only 37 blocks (see Figs. 13 and 14 for their decompositions). The compression ratio of various methods is summarized in Table 3. While PNG deflate, DT, GDM and FER methods efficiently employs the rectangular structure of the Labyrinth, the other methods (namely QTD) are not able to do so.

3.3. Backward composition

The compression methods would be useless, if we would not be able to restore the original shape if requested. In many

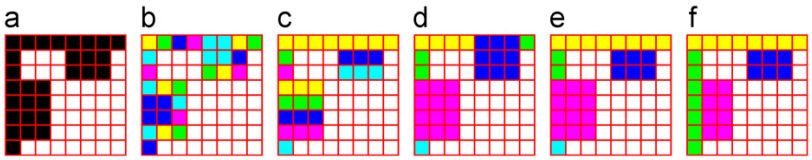


Fig. 12. Decomposition of a simple object: (a) original object—29 pixels, (b) QTD—23 blocks, (c) DM—10 blocks, (d) DT—6 blocks, (e) GDM—5 blocks, and (f) FER—4 blocks.

Table 2
Compression ratios on the LEAF database.

Format	Method	Size (byte)	ACR (%)
TIFF	No compression	172,886,448	100.00
TIFF	PackBits	13,282,998	7.68
TIFF	RLE	8,297,340	4.80
GIF	LZW	6,914,637	4.00
BLK	DT	5,173,371	2.99
PNG	Deflate	5,066,019	2.93
BLK	GDM	4,723,166	2.73
BLK	FER	4,603,942	2.66
BLK	QTD	3,012,532	1.74

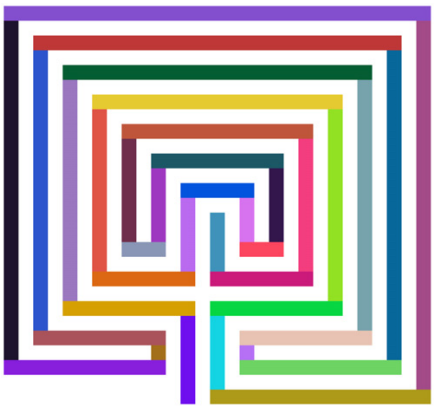


Fig. 13. Decomposition of the Labyrinth. DT, GDM, and FER produce the same number of 37 blocks.

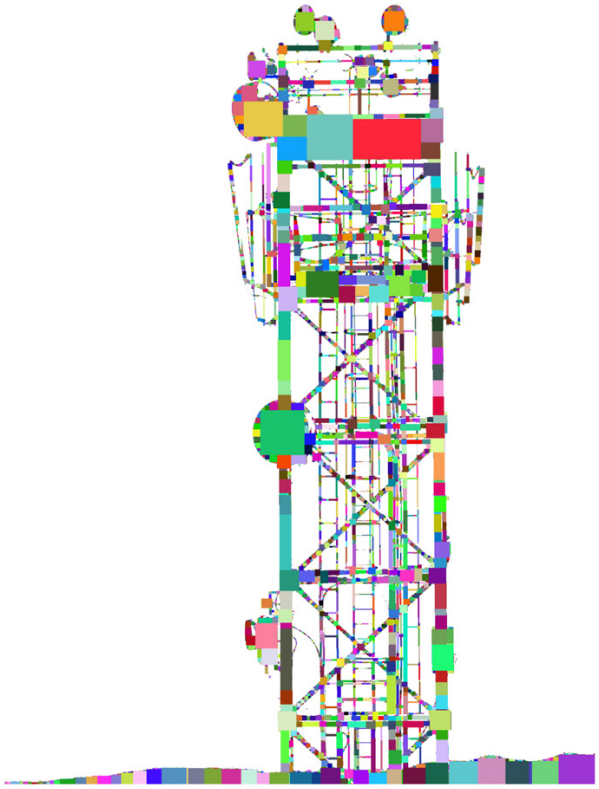


Fig. 14. Decomposition of the Transmitter.

applications the compressed images are stored in widely-accessed databases. While decomposition and compression is performed mostly off-line only once, composition (reconstruction) of the objects is required repeatedly and (almost) real time. Hence, three basic constraints on the reconstruction algorithm are no information loss, high speed and low memory consumption.

A naive reconstruction consists just of drawing all the rectangles to an allocated pixel array. This is accurate, simple to implement and relatively fast, but such method consumes a large amount of memory in an inefficient way. A much better way is to avoid drawing of the whole object and reconstruct the object boundary only. Since each binary object is fully determined by its boundary, this is still a lossless reconstruction. Such an algorithm handles computer memory more efficiently, because it does not require an array of the size of the original image.

We introduce an algorithm that requires only $\mathcal{O}(m+n)$ memory for an $m \times n$ image. Its time complexity varies with the shape of the image, i.e. with the number of the blocks K in the compressed version, but it performs very fast for "standard" objects. The algorithm works with images in BLK format regardless of the particular decomposition method used.

Table 3
Compression ratios of Transmitter and Labyrinth.

Format	Method	Transmitter (%)	Labyrinth (%)
TIFF	No comp.	100.00	100.00
TIFF	PackBits	24.01	49.65
TIFF	RLE	11.95	23.49
GIF	LZW	13.66	16.04
PNG	Deflate	6.29	1.43
BLK	DT	8.18	0.54
BLK	GDM	7.35	0.54
BLK	QTD	9.28	17.40
BLK	FER	6.66	0.54

The idea of the algorithm is to search the compressed BLK file and to maintain lists of border lines that we update whenever new rectangle is inserted. We start with the first rectangle of the compressed representation, so in the first step, the object boundary is formed by border lines of the first rectangle only. When processing a new rectangle, we need to update the current outline by inserting or removing lines or its parts. If the new

rectangle is adjacent to the current boundary, we insert only those parts of its border that are not adjacent to the current boundary. Moreover, in the current boundary, those lines or its parts that are adjacent to the new rectangle are removed (see Fig. 15).

To find adjacent border lines quickly, we maintain two arrays of lists, one for every horizontal coordinate and the other for every vertical one, see Fig. 16. Each element of the array contains pointer to the list of lines in the corresponding row or column. The borders of the new rectangle are inserted into the four corresponding lists and then the adjacent lists are searched.

First, we tested the correctness of the algorithm. We restored all leaves from BLK format to the standard matrix representation by the algorithm described above and compared them with the originals by matrix XOR. The cumulative error over the whole database was zero, which illustrates a perfect reconstruction. Also the reconstruction of complex objects is error free (see Fig. 17 for a reconstruction of a part of the Transmitter).

In the next experiment, we compared time complexity and memory consumption of the proposed decompression method with a "traditional" restoration, i.e. with putting all rectangles

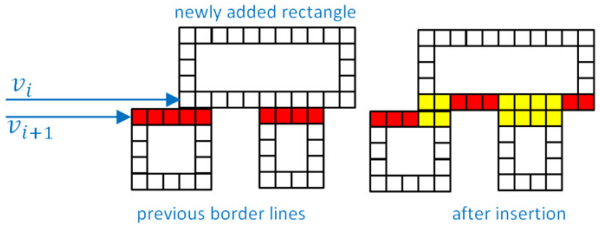


Fig. 15. Boundary list updating. Symbol v_i is the list of lines in the ith row.

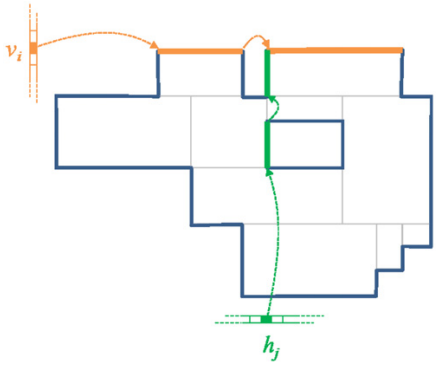


Fig. 16. Horizontal and vertical block boundaries. Symbols v_i and h_j are the list of lines in the ith row and jth column.

Fig. 17. Detail of the reconstruction of the Transmitter.

Table 4
Comparison of the proposed algorithm and the traditional reconstruction.

Criterion	Lower 10%	Upper 10%	Overall
Time (ms/image)			
Proposed	1.3	2.3	1.6
Traditional	4.0	4.0	4.0
Memory consumption (bytes/image)			
Proposed	4955	41,165	17,589
Traditional	1,048,576	1,048,576	1,048,576
Ratio (%)	0.5	3.9	1.7

into an array and consequent boundary detection by convolution with $(-1,1)$ kernel. The results achieved on the LEAF dataset are in Table 4. The column "Lower 10%" means 10% of objects with the lowest number of blocks; analogously the column "Upper 10%". "Overall" means a mean value over a complete database. One can see that we achieved huge savings in memory consumption and that also certain speed-up of the reconstruction can be observed. Clearly, both these factors are more significant for objects with a low number of blocks.

4. Experimental comparison—moment calculation

Moments are scalar quantities that have been used to characterize an image and to capture its significant features. From the mathematical point of view, moments are "projections" of an image function onto a polynomial basis. Functions of moments, insensitive to certain group of transforms, are called *moment invariants* (see [27] for a survey). Moment invariants have become one of the most important and most frequently used tools for object description and recognition. Hence, efficient algorithms of their computation are of a high importance and have attracted much attention (see [27, Chapter 7] for an overview).

Geometric moment of a continuous image $f(x,y)$ is defined as

$$m_{pq}^{(f)} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x,y)\,\mathrm{d}x\,\mathrm{d}y, \tag{2}$$

where $p+q$ is the order of the moment. If the image $f(x,y)$ is a discrete one of the size $M\times N$, then we can estimate its moment as[2]

$$\overline{m}_{pq}^{(f)} = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} i^p j^q f_{ij}. \tag{3}$$

For a binary object, f is just its characteristic function.

Object decomposition can be employed in moment calculation in the following way. If we decompose an object B into disjoint blocks $B_1,B_2,\ldots,B_K$ such that $B=\bigcup_{k=1}^{K} B_k$, then, thanks to the linearity of moments,

$$m_{pq}^{(B)} = \sum_{k=1}^{K} m_{pq}^{(B_k)}. \tag{4}$$

Since we can calculate the moment of each rectangular block in $\mathcal{O}(1)$ time (either by symbolic integration of the kernel function or, in a discrete domain, by summation rules), the overall complexity of $m_{pq}^{(B)}$ is $\mathcal{O}(K)$. If $K \ll MN$ the speed-up may be significant. As we already have seen, simple decomposition algorithms produce relatively high number of blocks but perform fast, while more sophisticated decomposition methods end up with small number of blocks but require more time. The

[2] Another way is using higher-order approximation of the integral and/or exact integration of $x^p y^q$ over rectangular regions [6].

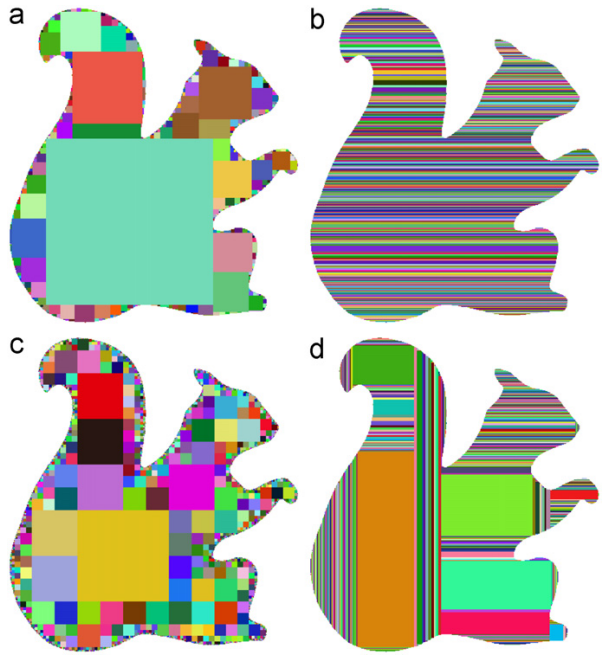


Fig. 18. Decomposition of the squirrel image: (a) DT—697 blocks, (b) GDM—497 blocks, (c) QTD—2513 blocks and (d) FER—476 blocks.

complexity of the decomposition must be always considered as a part of the whole algorithm. Even if the decomposition is performed only once and can be used for the calculation of all moments, the time needed for decomposing the image is often so long that it substantially influences the efficiency of the whole method.

We decomposed a 465×465 squirrel silhouette by means of DT, GDM, QTD and FER decompositions (see Fig. 18a–d) which lead to 697, 497, 2513, and 476 rectangular blocks and took 32, 1, 4, and 19 ms, respectively. Then we calculated the moments of the image up to the order 464. To prevent floating point overflow for high orders, we used discrete Chebyshev polynomials $\tau_p(x)$ instead of x^p in the kernel functions. The particular choice of the polynomial basis does influence neither the decomposition algorithms themselves nor their mutual comparison. The moments of the individual rectangles were calculated using summation rules of the kernel functions independently of the size of the rectangle.

The computation times are on the graph in Fig. 19, where the horizontal axis shows the number of moments calculated. The initial time $t(0)$ shows the expense of the decomposition. It can be better seen in more detailed Fig. 20.

The best results were achieved for GDM because of its fast initial decomposition time and only slightly sub-optimal number of blocks. The theoretically optimal FER method requires so much initial time that it produces best results only if we calculate about 45,000 and more moments, which is not realistic. QTD is the second fastest at the beginning, but the time grows quickly, as the number of moments increases such that it becomes the worst one, if 3000 and more moments are calculated. DT requires the longest initialization and then its time complexity grows faster than that of FER and GDM. It gave the worst results among the four tested methods in this experiment. All decomposition methods outperformed the calculation from the definition even for very low number of moments.

The above results can be generalized for most "reasonable simple" shapes. However, it is easy to find a counterexample.

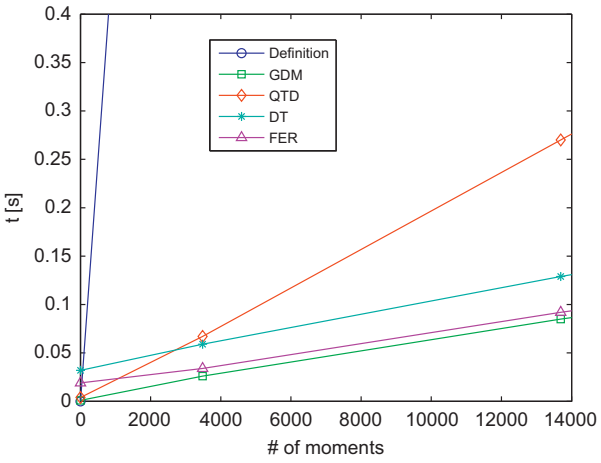


Fig. 19. The time complexity of the moments of the squirrel image.

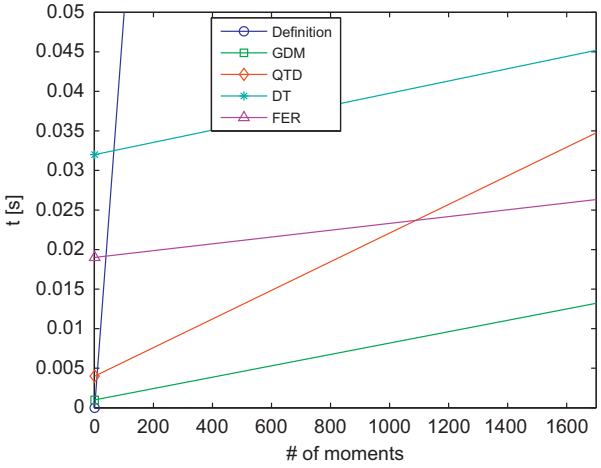


Fig. 20. A detail of Fig. 19 showing the complexity of lower-order moments.

If we repeat this experiment for a chessboard image, the results change dramatically, see Fig. 21. The chessboard image is the worst possible case, because it cannot be decomposed efficiently. The decomposition is just a waste of time and a direct calculation from the definition exhibits the best performance.

5. Experimental comparison—convolution

In this experiment, we show that the decomposition can be a powerful tool for speeding-up convolution filtering of an image with a binary kernel. While traditional "by definition" discrete convolution of an $M \times N$ image with an arbitrary $m \times n$ mask requires $\mathcal{O}(mnMN)$ operations and convolution via fast Fourier transform (FFT) $\mathcal{O}(MN \log MN)$ operations, convolution with a constant-valued rectangle takes only $\mathcal{O}(MN)$ operations regardless of the mask size. There are several ways how to implement such algorithm. The best one is probably to employ the precalculations of row-wise and consequently column-wise sums of the image. The convolution in a certain position is then calculated just from four values, which corresponds to the mask corners (see Fig. 22).

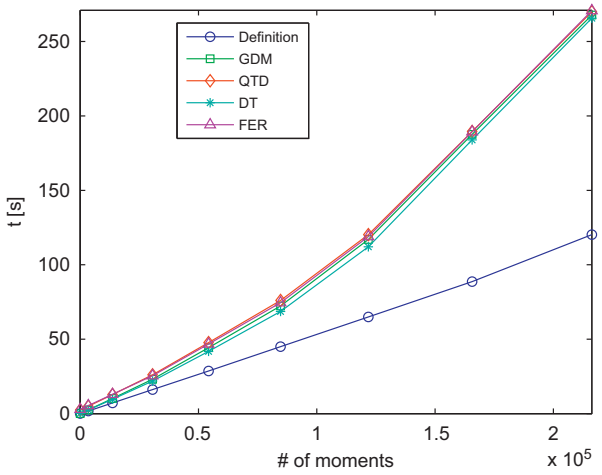


Fig. 21. The time complexity of the moments of the chessboard image.

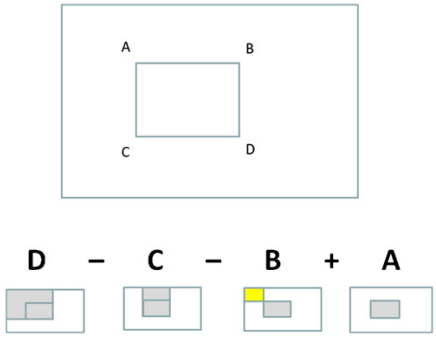


Fig. 22. Convolution of an image with a constant rectangular mask ABCD. The large rectangle contains in point X row-wise and column-wise sum $S(X)$ of the original image from (0, 0) to X. Then the convolution in the depicted position is given as $S(D)-S(C)-S(B)+S(A)$.

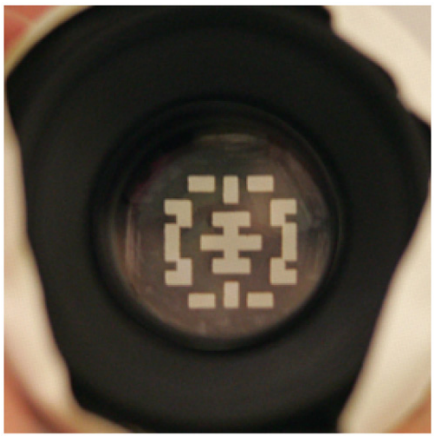


Fig. 23. An example of the coded aperture mask (courtesy of [29]).

Now, let us imagine a binary kernel, where the non-zero values form a more complex set than a single rectangle. Such a situation typically appears for instance in *coded aperture* (CA) imaging. CA is a method of recovering the depth map of the scene from a single image [28]. The trick is to insert a special occluder within the aperture of the camera lens to create a coded aperture (see Fig. 23 for an example). The CA output must be deconvolved, which incorporates (if an iterative deconvolution method such as Richardson–Lucy or similar is applied) repeating the calculations of convolution of an estimated image with the aperture mask. If the mask has a full or close-to-full rank, we cannot effectively use any factorization, which seemingly takes us back to the calculation from the definition or via FFT. However, if we decompose the mask into rectangles $B_1,\ldots,B_K$, we can, thanks to the linearity of convolution, calculate the convolutions with each B_j separately by the method described above and then just sum up the results. (This method can be used even if the kernel is not binary and thus the blocks have different values but that is a rare case in practice.) In that way, we achieve the overall complexity of $\mathcal{O}(KMN)$. In this experiment, we demonstrate that for $K \ll mn$ the speed up is really huge comparing to the direct calculation from the definition and still significant comparing to the FFT.

We filtered a 3456×2592 image with two binary kernels of the same shape but different size—the small one of (35×38) and the large one of (141×152) pixels. The small kernel is scaled version of the large one. The kernels were decomposed by the FER method into 10 rectangles (see Fig. 24 for the kernel and its decomposition). We tested three methods—direct convolution in the image domain from the definition, convolution via FFT in the frequency domain (we used popular public-domain FFTPack software [30,31]), and fast convolution using kernel decomposition as described above. In the last case, the matrix of the partial sums of the image was precomputed and then the (cyclic) convolution was calculated as explained in Fig. 22. We wanted to measure the time of each individual step separately, because in practice, either the mask decomposition or the precomputing of partial sums uses to be done only once and hence its complexity is negligible (in batch processing either the mask or the image stays the same while the other factor varies). However, the mask decomposition was so fast that the corresponding time was not measurable.

The smoothed image is always the same regardless of the method used. The time comparison is summarized in Table 5. As expected, the slowest calculation is from the definition in the image domain. Even for the small mask, it is many times slower than the methods via FFT and via mask decomposition. Note that the times for both FFT and decomposition methods actually do not depend on the mask size, which is clear from the theory. Although FFTPack is a very powerful implementation, the decomposition method was still able to perform four times faster. Precomputing of the partial sums took only 0.1 s of the total time.

Fig. 24. The convolution kernel used in the experiment (left) and its FER decomposition into 10 blocks (right).

Table 5
Time comparison of various convolution methods (the time in s).

Mask size	Definition	FFT	Decomposition
35×38	26	4.3	0.96
141×152	411	4.3	0.96

This experiment illustrates that the convolution with a binary mask can be implemented by means of mask decomposition in a very efficient way. Two main factors influence whether or not the convolution via decomposition is faster than via FFT—the image size (not the mask size!) and the number of blocks of the mask. In our implementation and hardware and for 8–10 Mpix images the threshold value is about 40. If the number of blocks is lower, the decomposition-based convolution is the best choice.

6. Conclusion

We presented an overview of methods which decompose an arbitrary binary object into rectilinear rectangles, starting from very simple one up to the optimal graph-based decomposition. We tested their performance in three frequent tasks—image compression, moment computation and linear filtering. We showed that there is no "generally best" method; the choice must reflect our requirements and is always a compromise between complexity on one hand and time and memory consumption on the other hand. The weights given to these two factors are user-defined parameters. This paper should help the users to select proper decomposition method according to their preferences. In our opinion, GDM is the most appropriate in common situations, while FER is recommended, if we want to achieve as few blocks as possible on the expense of higher complexity. The other two tested methods either produce too many blocks (QTD) or perform slowly (DT). They may find applications in specific tasks only.

An interesting extension in the future could be a usage of overlapping blocks (we speak about *covering* instead of partitioning). This may significantly decrease the block number, however, on the expense of the NP-hard complexity.

Although we have been talking just about binary objects in the paper, all methods can be theoretically used for graylevel and color images as well. Graylevel image can be expressed as a union of disjoint binary images, which can be obtained either as intensity slices [32] or bit planes [33]. However, these "images" use to be highly fragmented (especially low bit planes resemble a "random chessboard") and decomposition methods do not perform well. Our experiments indicate that for graylevel/color images the decomposition algorithms of this kind have only little practical importance.

Acknowledgment

This work has been supported by the Grant no. P103/11/1552 of the Czech Science Foundation.

References

[1] J.M. Keil, Polygon decomposition, in: Handbook of Computational Geometry, Elsevier, 2000, pp. 491–518.
[2] M.F. Zakaria, L.J. Vroomen, P. Zsombor-Murray, J.M. van Kessel, Fast algorithm for the computation of moment invariants, Pattern Recognition 20 (6) (1987) 639–643.
[3] M. Dai, P. Baylou, M. Najim, An efficient algorithm for computation of shape moments from run-length codes or chain codes, Pattern Recognition 25 (10) (1992) 1119–1128.
[4] B.C. Li, A new computation of geometric moments, Pattern Recognition 26 (1) (1993) 109–113.
[5] I.M. Spiliotis, B.G. Mertzios, Real-time computation of two-dimensional moments on binary images using image block representation, IEEE Transactions on Image Processing 7 (11) (1998) 1609–1615.
[6] J. Flusser, Refined moment calculation using image block representation, IEEE Transactions on Image Processing 9 (11) (2000) 1977–1978.
[7] C.-H. Wu, S.-J. Horng, P.-Z. Lee, A new computation of shape moments via quadtree decomposition, Pattern Recognition 34 (7) (2001) 1319–1330.
[8] J.H. Sossa-Azuela, C. Yáñez-Márquez, J.L. Díaz de León Santiago, Computing geometric moments using morphological erosions, Pattern Recognition 34 (2) (2001) 271–276.
[9] T. Suk, J. Flusser, Refined morphological methods of moment computation, in: 20th International Conference on Pattern Recognition ICPR'10, IEEE Computer Society, 2010, pp. 966–970.
[10] W. Liou, J. Tan, R. Lee, Minimum partitioning simple rectilinear polygons in $O(n \log \log n)$-time, in: Proceeding of the 5th Annual ACM Symposium on Computational Geometry SoCG'89, ACM, New York, NY, USA, 1989, pp. 344–353.
[11] K.D. Gourley, D.M. Green, A polygon-to-rectangle conversion algorithm, IEEE Computer Graphics and Applications 3 (1) (1983) 31–36.
[12] E. Kawaguchi, T. Endo, On a method of binary-picture representation and its application to data compression, IEEE Transactions on Pattern Analysis and Machine Intelligence 2 (1) (1980) 27–35.
[13] J. Flusser, An adaptive method for image registration, Pattern Recognition 25 (1) (1992) 45–54.
[14] D.N. Vizireanu, Generalizations of binary morphological shape decomposition, Journal of Electronic Imaging 16 (1) (2007) 013002-1-6.
[15] G. Borgefors, Distance transformations in digital images, Computer Vision, Graphics, and Image Processing 34 (3) (1986) 344–371.
[16] M. Seaidoun, A Fast Exact Euclidean Distance Transform with Application to Computer Vision and Digital Image Processing, Ph.D. Thesis, Northeastern University, Boston, USA, Advisor John Gauch, September, 1993.
[17] T.E. Schouten, E.L. van den Broek, Incremental distance transforms (IDT), in: 20th International Conference on Pattern Recognition ICPR'10, IEEE Computer Society, 2010, pp. 237–240.
[18] W. Lipski Jr., E. Lodi, F. Luccio, C. Mugnai, L. Pagli, On two-dimensional data organization II, in: Fundamenta Informaticae, Annales Societatis Mathematicae Polonae, Series IV, vol. II, 1979, pp. 245–260.
[19] T. Ohtsuki, Minimum dissection of rectilinear regions, in: Proceedings of the IEEE International Conference on Circuits and Systems ISCAS'82, IEEE, 1982, pp. 1210–1213.
[20] L. Ferrari, P.V. Sankar, J. Sklansky, Minimal rectangular partitions of digitized blobs, Computer Vision, Graphics, and Image Processing 28 (1) (1984) 58–71.
[21] H. Imai, T. Asano, Efficient algorithms for geometric graph search problems, SIAM Journal on Computing 15 (2) (1986) 478–494.
[22] D. Eppstein, Graph-theoretic solutions to computational geometry problems, in: 35th International Workshop on Graph-Theoretic Concepts in Computer Science WG'09, Lecture Notes in Computer Science, vol. 5911, Springer, 2009, pp. 1–16.
[23] A.V. Goldberg, S. Rao, Beyond the flow decomposition barrier, Journal of the Association for Computing Machinery 45 (5) (1998) 783–797.
[24] A.V. Goldberg, R.E. Tarjan, A new approach to the maximum-flow problem, Journal of the Association for Computing Machinery 35 (4) (1988) 921–940.
[25] J. Edmonds, R.M. Karp, Theoretical improvements in algorithmic efficiency for network flow problems, Journal of the Association for Computing Machinery 19 (2) (1972) 248–264.
[26] Department of Image Processing, Tree LEAF database, URL ⟨http://zoi.utia.cas.cz/tree_leaves⟩.
[27] J. Flusser, T. Suk, B. Zitová, Moments and Moment Invariants in Pattern Recognition, Wiley, Chichester, 2009.
[28] A. Levin, R. Fergus, F. Durand, W.T. Freeman, Image and depth from a conventional camera with a coded aperture, in: Special Interest Group on Computer Graphics and Interactive Techniques Conference SIGGRAPH'07, ACM, New York, NY, USA, 2007.
[29] A. Levin, R. Fergus, F. Durand, B. Freeman, Image and depth from a conventional camera with a coded aperture, URL ⟨http://groups.csail.mit.edu/graphics/CodedAperture⟩.
[30] P.N. Swarztrauber, FFTPACK, URL ⟨http://www.netlib.org/fftpack/⟩.
[31] A. Fernandes, FFTPACK translated to pure iso C/C++, URL ⟨http://www.fernandes.org/txp/article/4/fftpack-translated-to-pure-iso-cc⟩.
[32] G.A. Papakostas, E.G. Karakasis, D.E. Koulouriotis, Efficient and accurate computation of geometric moments on gray-scale images, Pattern Recognition 41 (6) (2008) 1895–1904.
[33] I.M. Spiliotis, Y.S. Boutalis, Parameterized real-time moment computation on gray images using block techniques, Journal of Real-Time Image Processing (2011) 81–91.

Tomáš Suk received the MSc degree in electrical engineering from the Czech Technical University, Faculty of Electrical Engineering, Prague, 1987. The CSc degree (corresponds to PhD) in computer science from the Czechoslovak Academy of Sciences, Prague, 1992. From 1991 he is a researcher with the Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Prague, a member of Department of Image Processing. In 2002 he received the Otto Wichterle's premium of the Academy of Sciences of the Czech Republic for young scientists.

His research interests include digital image processing, pattern recognition, image filtering, invariant features, moment and point invariants, geometric transformations of images. Applications in botany, remote sensing, astronomy, medicine and computer vision. He has published 18 papers in international journals and more than 50 conference papers, mostly from international conferences. He is a coauthor of the monograph "Moments and Moment Invariants in Pattern Recognition" (Wiley, 2009).

Cyril Höschl IV received the MSc degree in computer science from the Charles University, Faculty of Mathematics and Physics, Prague, 2010. Now he is a PhD student in the Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Prague. His supervisor is Jan Flusser.

His research interests include digital image processing. Among this area he is author of algorithms and software for visualization of relations known as Sociomapping.

Jan Flusser received the MSc degree in mathematical engineering from the Czech Technical University, Prague, Czech Republic in 1985, the PhD degree in computer science from the Czechoslovak Academy of Sciences in 1990, and the DSc degree in technical cybernetics in 2001. Since 1985 he has been with the Institute of Information Theory and Automation, Academy of Sciences of the Czech Republic, Prague. In 1995–2007, he was holding the position of a head of Department of Image Processing. Currently (since 2007) he is a director of the Institute. He is a full professor of computer science at the Czech Technical University and at the Charles University, Prague, Czech Republic, where he gives undergraduate and graduate courses on digital image processing, pattern recognition, and moment invariants and wavelets.

His research interest covers moments and moment invariants, image registration, image fusion, multichannel blind deconvolution, and super-resolution imaging. He has authored and coauthored more than 150 research publications in these areas, including the monograph "Moments and Moment Invariants in Pattern Recognition" (Wiley, 2009), tutorials and invited/keynote talks at major international conferences.

He is a senior member of the IEEE.

Close-to-optimal algorithm for rectangular decomposition of 3D shapes

Cyril Höschl IV and Jan Flusser

Institute of Information Theory and Automation of the CAS, Pod vodárenskou věží 4, 182 08 Praha 8, Czech Republic

Abstract

In this paper we propose a novel algorithm for a decomposition of 3D binary shapes to rectangular blocks. The aim is to minimize the number of blocks. Theoretically optimal brute-force algorithm is known to be NP-hard and practically infeasible. We introduce its sub-optimal polynomial heuristic approximation, which transforms the decomposition problem onto a graph-theoretical problem. We compare its performance with the state of the art Octree and Delta methods. We show by extensive experiments that the proposed method outperforms the existing ones in terms of the number of blocks on statistically significant level. We also discuss potential applications of the method in image processing.

Keywords: 3D binary object, voxels, decomposition, rectangular blocks, sub-optimal algorithm, tripartite graph, maximum independent set

1. Introduction

Binary images, both in 2D and 3D, form a specific class of objects and require dedicated algorithms for their processing and analysis. The major difference from traditional gray-level and color images is that the pixel/voxel matrix representation of binary images (which consists only of zeros and ones) is highly redundant. This has led to many specialized algorithms that employ various

*Corresponding author
Email address: hoschl@utia.cas.cz (Cyril Höschl IV and Jan Flusser)

loss-less compressive representations for image storage and object description (see, for instance, the books [1], [2]). Such representations result not only in an efficient memory usage but also contribute to fast feature calculation and object recognition.

One of the possible approaches (and probably the most frequently used one) is to *decompose* the object into simple parts which we are able to store and process efficiently (some other approaches, such as object characterization based on its boundary and various kinds of multilevel representations, exist but are beyond the scope of this paper). Having a binary object B (by a binary object we understand a set of all pixels of a binary image whose values equal one), we decompose it into $K \geq 1$ partitions $B_1, B_2, \ldots, B_K$ such that $B_i \cap B_j = \emptyset$ for any $i \neq j$ and $B = \bigcup_{k=1}^{K} B_k$.

The 2D decomposition problem has been studied for decades in computational geometry and some of the methods were later introduced into the image analysis area. Although in the continuous domain we may consider various shapes of the partitions (convex, star-shaped, hexagonal, rectangular, etc., see [3]), the decomposition methods in the discrete domain should use only rectilinear rectangular blocks because of a native rectangular structure of the discrete image domain (if other primitives were allowed, we would have to face sampling errors along the boundary).

A commonly accepted measure of the decomposition quality is the number of the resulting blocks K. This is a reasonable criterion, justified by the fact that the complexity of subsequent calculations uses to be $\mathcal{O}(K)$ and compression ratio (if the decomposition is used for compression purposes) also increases as the number of blocks decreases. The time complexity of the decomposition is usually the secondary criterion. Obviously, sophisticated decomposition methods which end up with small number of blocks usually require more time than the simple ones. Since the decomposition is in most tasks performed only once per object and can be done off-line, the time complexity becomes crucial only if it is so

high that the method is not feasible in an acceptable time.[1]

Several rectangular 2D decomposition algorithms have been proposed namely in connection with compression and image feature calculation [4, 5, 6, 7, 8, 9, 10, 11] but one may find also other applications in fast spatial filtering, in iterative deconvolution methods [12], in the coded aperture imaging [13], in integrated circuits design [14, 15], and in other areas. The decomposition methods in the above cited papers are simple, intuitive but only suboptimal – they do not guarantee the minimal number of the blocks. In computational geometry, several authors [16, 17, 18] independently proposed basically the same algorithm (later discussed and improved in [19, 20]) which was proved to be optimal since it actually minimizes the number of blocks for an arbitrary input shape. The algorithm has a polynomial time complexity. This algorithm was adapted for image analysis purposes by Suk et al. [21], who also performed a large-scale experimental comparison with other methods. Their experiments proved not only the optimality but also an acceptable time-complexity of the algorithm (the time consumption was of the same order as in the case of the other methods) [21]. In this sense, the 2D decomposition problem has been fully resolved.

During the last decade, 3D image/object analysis has attracted a significant attention due to a dynamic development of 3D imaging devices and technologies. Most of the devices only measure the distance to the object surface, they cannot see "inside" the object.[2] The object is scanned from various sides such that each surface part is visible on at least one scan. The individual scans (views) are combined together and the complete surface of the object is reconstructed. We obtain 3D binary object as a result. All range finders, including the popular X-box sensor Kinect, work in that way. All popular public benchmark databases of 3D objects such as Princeton Shape Benchmark (PSB) and McGill 3D Shape Benchmark [22, 23] contain binary objects. This illustrates there is a great

[1] The term "acceptable" of course depends on the particular application.

[2] In this paper, we do not consider specialized medical devices such as MRI, CT, SPECT and PET, that actually produce a full 3D voxel cube.

demand of developing efficient algorithms for working with 3D binary objects.

Few papers on 3D shape decomposition can be found in the literature but almost none of them has solved the decomposition into rectangular blocks. Ren et al. dealt with a decomposition into "nearly convex" regions [24], Sivignon and Coeurjolly [25] decomposed the object surface into planar patches and several other authors proposed decompositions into geometric primitives and other predefined components [26, 27], which requires some kind of "understanding" the object structure.

To our best knowledge, the only paper on 3D shape decomposition into rectangular blocks is by Dielissen and Kaldewaij [28], who proved that decision problem of the optimal 3D decomposition (i.e. that one which minimizes K) is equivalent to a variant of the *Boolean three satisfiability* problem called 3SAT3. This means that the optimal 3D decomposition problem is NP-complete and cannot be efficiently resolved. Nevertheless, 3D decomposition can be accomplished by various sub-optimal methods. Some simple algorithms can be easily designed as an extension of 2D methods. Run-length encoding, the delta-method, and the quadtree decomposition (which turns into octree in 3D) are typical examples.

In this paper, we present a new sub-optimal algorithm. It was inspired by the optimal 2D decomposition algorithm [18, 21] but unlike the optimal 3D algorithm the proposed method is of a polynomial complexity. From this point of view, it can be considered a polynomial approximation[3] for an NP-complete algorithm. As demonstrated experimentally and by statistical tests, the proposed method outperforms both delta-method and octree decomposition significantly.

[3] To the best of our knowledge, no approximation that provably bounds the optimal solution up to a small constant factor is known. In this text, by an approximation we mean a polynomial heuristic that provides reasonable good solution compared to the NP-hard optimal algorithm.

2. Present state of the art

Since 3D decomposition methods are mostly motivated by their 2D ancestors[4], we start our brief review with a description of 2D versions. Then we explain how to extend the methods into 3D.

2.1. Delta method

The Delta method (DM) was under this name originally proposed by Zakaria [4] in the connection with computation of object descriptors but it has been known also in image compression as the run-length encoding (RLE). In the basic version, the "blocks" are continuous row segments for which only the coordinate of the beginning and the length are stored. The method was later slightly improved by Dai [5] and generalized for non-convex shapes by Li [6]. The Delta method is very fast but leads to the number of blocks which uses to be (much) higher than the minimal decomposition. A simple but powerful improvement of the delta method was proposed by Spiliotis and Mertzios [7] and improved later by Flusser [8]. In this "Generalized Delta method" (GDM), the adjacent rows are compared and if there are some with the same beginning and end, they are unified to form a rectangle. The GDM is only slightly slower than the basic DM while producing (sometimes significantly) less number of blocks.

Creating a 3D version of DM is straightforward. In case of the generalized Delta method, the 3D version (denoted as 3GDM) is also clear but we have to test which direction and which order of the segment connecting yields the best result. In 2D we have only two degrees of freedom – we may choose either the vertical or the horizontal direction. As soon as the direction has been chosen, there is no choice in the row connection step. In 3D, we have an option of any of three directions and for each direction we may first connect horizontally adjacent rows and then vertically adjacent "plates" or vice versa.[5] Hence, the 3GDM which we implemented and used in the tests in this paper checks all

[4] We are not aware of any 3D method which does not have a counterpart in 2D

[5] Here the terms "horizontal" and "vertical" are used relatively to the chosen direction.

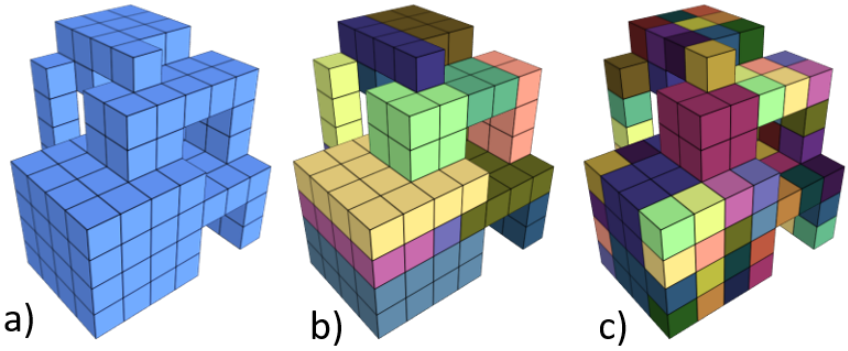


Figure 1: A sample object (a), its decomposition by the 3GDM into 13 blocks (b), and the octree decomposition into 90 blocks (c).

these six options and selects that one yielding the minimum number of blocks or decides randomly if there is a multiple minimum (see Fig. 1b for an example of the 3GDM decomposition).

2.2. Quadtree and octree decompositions

In 2D, the Quadtree decomposition (QTD) is a popular hierarchical decomposition scheme used in several image processing areas including representation and compression [29], spatial transforms [30] and feature calculation [9]. In its basic version, the QTD works with square images of a size of a power of two. If this is not the case, the image is zero-padded to the nearest such size. The image is iteratively divided into four quadrants. Homogeneity of each quadrant is checked and if the whole quadrant lies either in the object or in the background it is not further divided. If it contains both object and background pixels, it is divided into quadrants and the process is repeated until all blocks become homogeneous. The decomposition can be efficiently encoded into three-symbol string, where 2 means division, 1 means a part of the object and 0 stands for the background.

The QTD always yields square blocks which may be advantageous for some purpose but usually leads to a much higher number of blocks than necessary. It

would be possible to implement a backtracking and to unify the adjacent blocks of the same size into a rectangle but this would increase the complexity. Since the speed is the main advantage of this method, the backtracking is mostly not employed here. A drawback of this decomposition algorithm is that the division scheme is not adapted with respect to the content of the image but it is defined by absolute spatial coordinates. Hence, the decomposition is not translation-invariant and may lead to absurd results when for instance a large single square is uselessly decomposed up to individual pixels.

The QTD can be readily extended into 3D. Instead of square elements we employ cubes and the quadtree scheme is replaced with the octree (OTD). The OTD method keeps all pros and cons of its 2D ancestor (see Fig. 1c for an example of the OTD decomposition).

Octree representation has been commonly used in Minecraft-like computer games and similar voxel-represented scenarios. Popularity of this representation is mainly due to its simple implementation, fast run, and its ability to render the scene in various levels of details. However, it is much less efficient than other methods in terms of the block number.

2.3. Binary space partitioning

The term *Binary space partitioning* (BSP) denotes a wide class of hierarchical decomposition methods, very popular namely in 3D computer graphics [31, 32]. BSP recursively divides the bounding box, which contains the object, by hyperplanes. Individual BSP methods differ from one another by the criteria that select the separating hyperplane. Both QTD and OTD are special cases of BSP, where the hyperplanes are always put in the middle of the box to be decomposed. Other BSP algorithms may put the separator with regard of the object itself, but the separators always cut the entire box into two parts. This leads to object representation by the BSP tree. The BSP algorithms usually perform fast but the number of blocks depends on the chosen criterion.

2.4. Optimal decomposition

As we already pointed out, there exist a 2D decomposition method of polynomial complexity (even in several versions) which guarantees the minimum number of blocks for an arbitrary shape. Here we briefly recall the version proposed in [21].

The method performs hierarchically on two levels. On the first level, we detect all "concave" vertices (i.e. those having the inner angle 270°) of the input object and identify pairs of "cogrid" concave vertices (i.e. those having the same horizontal or vertical coordinates). Then we divide the object into subpolygons by constructing chords which always connect two cogrid concave vertices. As proved in [18] and in other papers, the optimal choice of the chord set is such that the chords are pair-wise disjoint and their number is maximum possible.

The problem of optimal selection of the chords is equivalent to the problem of finding the maximal set of independent vertices in a graph, where each vertex corresponds to a chord and two vertices are connected by an edge if the two chords have a common point (either a concave vertex or an intersection). Generally, this problem is NP-complete, but our graph is a bipartite one, since any two horizontal (vertical) chords cannot intersect one another. In a bipartite graph, this task can be efficiently resolved. We find a maximal matching, which is a classical problem in graph theory, whose algorithmic solution in a polynomial time has been published in various versions. Some of them are optimized with respect to the number of edges, the others with respect to the number of the vertices (see [33, 34, 35, 19] for some examples of particular algorithms) but all of them are polynomial in both.

As soon as the maximal matching has been constructed, the maximal set of independent vertices can be found much faster than the maximal matching itself – roughly speaking, the maximal independent set contains one vertex of each matching pair plus all isolated vertices plus some other vertices, which are not included in the matching but still independent. As a result, we obtain a set of vertices that is unique in terms of the number of vertices being involved but

ambiguous in terms of the particular vertices. However, this ambiguity does not matter – although each set leads to different object partition, the number of the components is always the same. Hence, at the end of the first level, the object is decomposed into subpolygons, which do not contain any cogrid concave vertices (see Fig. 2).

The second level is very simple. Each subpolygon arriving from the first level is either a rectangle or a concave polygon. In the latter case, it must be further divided. From each its concave vertex, a single chord is constructed such that this chord terminates either on the boundary of the subpolygon or on the chord that has been constructed earlier. This is a sequential process in which each concave vertex is visited only once. The order of the concave vertices may be chosen arbitrary. Similarly, we may choose randomly between two possible chords offered in each concave vertex. This choice does not influence the final number of blocks. After that, the subpolygon is divided into rectangles, because rectangle is the only polygon having no concave vertices.

The optimal decomposition cannot be readily extended into 3D because it becomes NP-complete, as follows from the analysis presented in the next Section. The method we propose in this paper replaces the NP-complete steps by an approximation of a polynomial complexity.

3. 3D suboptimal decomposition

When trying to extend the above 2D optimal algorithm into 3D, we discover several substantial differences between the 2D and 3D cases. In 3D, concave edges play the role of concave vertices (see Fig. 3). Concave vertices may exist, too, but they have no significance for 3D decomposition. The analogue of the chord is the *separator*, which is the intersection of a plane and the object (see Fig. 4). Note that the separator not always splits the entire object into two separate components as it is illustrated in Fig. 5. Similarly as in 2D, any concave edge must be contained in a separator to get the decomposition into blocks. Separators containing no concave edges are possible but useless. Unlike

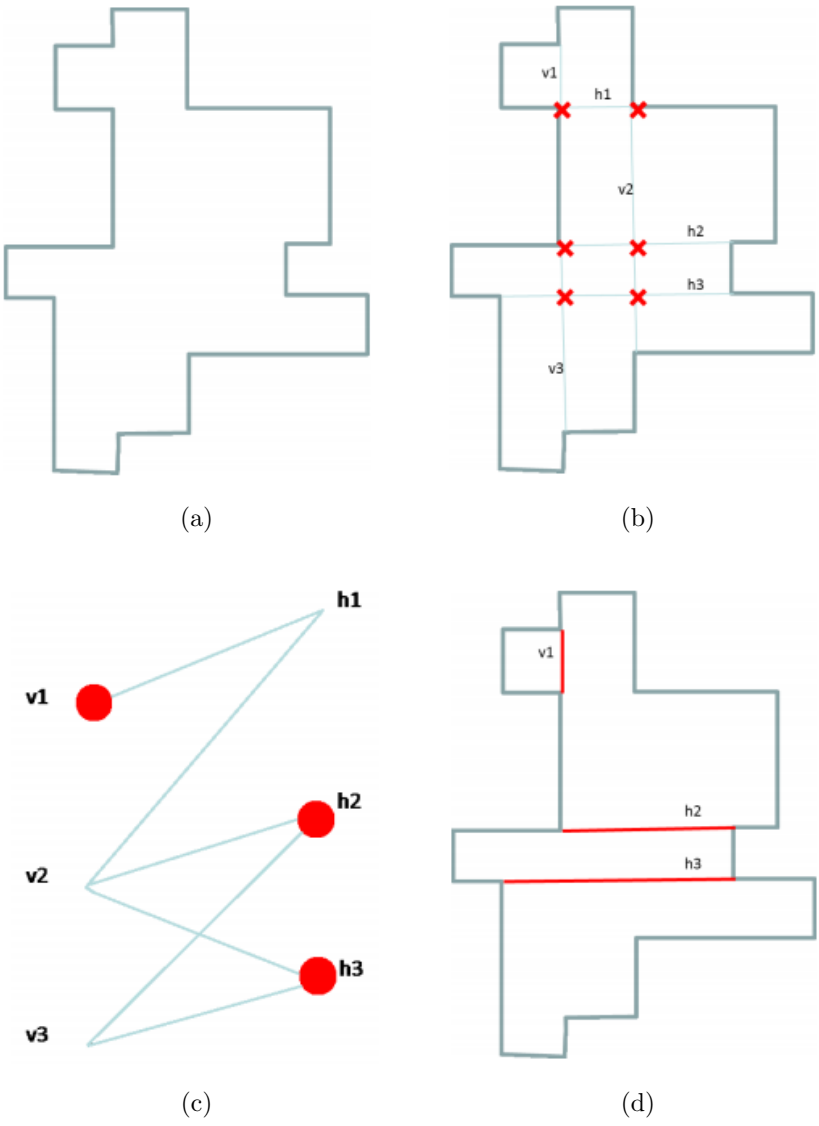


Figure 2: The first level of the 2D optimal decomposition method. (a) The input object. (b) All possible chords connecting two cogrid concave vertices. The crosses indicate the chord intersections. (c) The corresponding bipartite graph with a maximum independent set of three vertices. Other choices are also possible, such as $\{h_1, h_2, h_3\}$ or $\{v_1, v_2, v_3\}$. (d) The first level of the object decomposition.

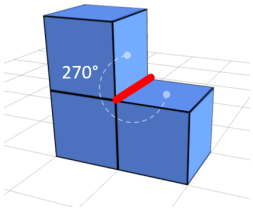


Figure 3: A concave edge formed by two voxel faces of a 270°angle in between.

2D, where a chord can contain two concave vertices at maximum, a separator can contain an *arbitrary high number* of concave cogrid edges (the edges laying in a plane which is perpendicular to an axis are called *cogrid edges*). From this we can see that the 3D version of the optimal algorithm (if it exists) cannot work in two levels but rather in m levels, where m depends on (but not necessarily equals to) the maximum number of the existing cogrid concave edges. Another difference from 2D is that a separator may split a perpendicular concave edge into two separate concave edges. In this way, placing a separator eliminates some concave edges but may at the same time induce new ones, which is impossible in 2D (see Fig. 6). The most significant difference is, however, the following one. Even if we place the separators in order given by the number of the concave edges they eliminate, we do not end up with the minimum number of blocks. Placing a separator which eliminates the maximum possible number of the concave edges at the particular moment may not be globally optimal since it may prevent placing some separator(s) which would finally lead to a better decomposition (see Fig. 8 for illustration of such simple situation). Before we fix the separator, we should check the complete subtree of all other alternatives. This makes the task NP-complete.[6]

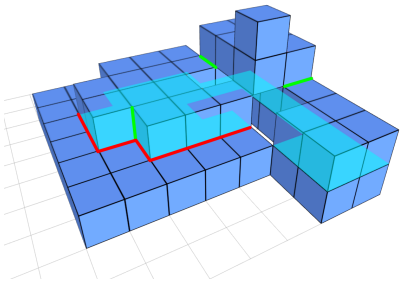


Figure 4: A separator is a cross-section of a plane and the object. Meaningful separators eliminate some concave edges of the object. In this example, the red edges have been eliminated by a cyan separator.

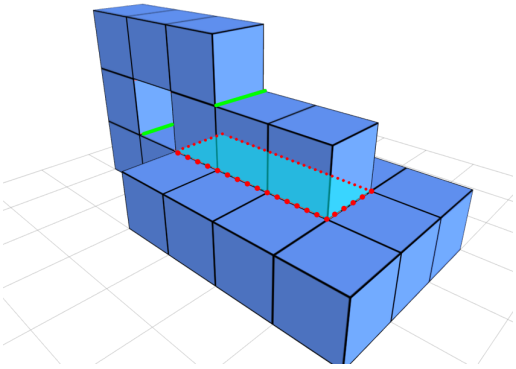


Figure 5: Example of a separator which is meaningful (it eliminates red dotted edges) even if it does not split the object into separated parts.

3.1. The basic version

The basic version of a sub-optimal algorithm is a heuristics which basically follows from the above thoughts. It works iteratively in a greedy manner. In each iteration, we place proper separators and cut the object by them. This eliminates all edges connected with the chosen separators. We repeat this step until all concave edges have disappeared.

The tricky part in each iteration is how to choose the proper separators. We have already shown that the optimal brute force approach is NP-complete. To overcome this, we choose the separators according their *weight*. The weight w_s of separator s is a function which estimates how significant (i.e. how useful) is the particular separator for the decomposition. Intuitively, it should reflect the number of the concave edges the separator eliminates and should be easy to evaluate (preferably in a polynomial time). Two possible particular choices of w_s will be discussed later.

In each iteration, the algorithm finds all possible separators and calculates their weights. Let us denote the highest weight as α and the set of all separators with this weight as M. Now the method tries to place as many separators from M as possible but at the same time it must avoid all mutually intersecting separators because they are redundant (by "intersecting" we understand also adjacent separators, i.e. those which share an edge), see Figs. 6 and 7. In other words, we are looking for a maximum subset of M of non-intersecting separators.

This task can be reformulated as a task of finding the maximum independent set in a tripartite graph, which is a well-known problem in graph theory. We refer to the maximum independent set in graph G as $MaxIS(G)$ or $MaxIS$ for short.

We construct graph $G = (V, E')$ whose vertices are the separators from the set M. Vertices u and v form edge $(u, v) \in E'$ iff the corresponding separators

[6]Note that the NP-completeness was formally proven in [28] by transforming the decomposition problem onto a 3SAT3 problem.

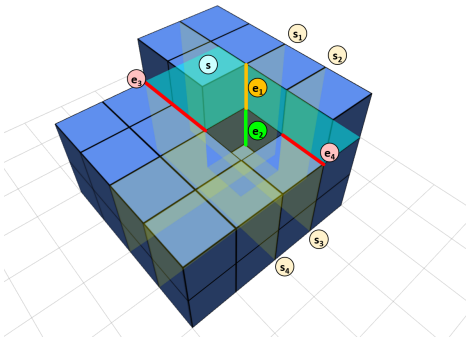


Figure 6: An example of a separator that splits perpendicular concave edges and intersects other separators. Cyan-highlighted separator s eliminates edges e_3, e_4 and intersects one vertical edge that has been divided into e_1 and e_2. It also intersects other separators s_1 and s_2 and is adjacent to separators s_3 and s_4

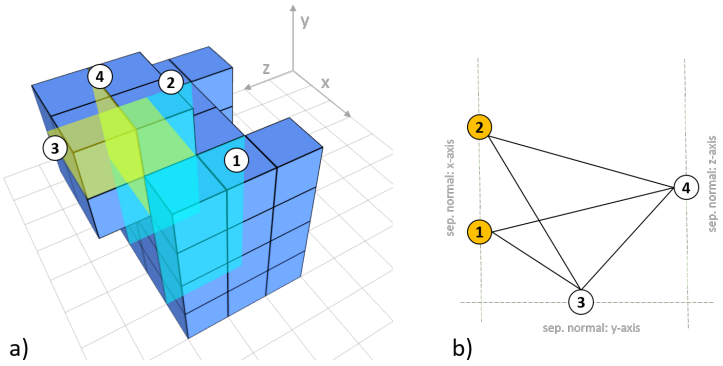


Figure 7: An example of a graph construction: On the left there are four separators of the same weight. On the right we can see the corresponding tripartite graph. The graph vertices are associated with the separators and the graph edges reflect their mutual intersections (or adjacency). In this example, highlighted vertices {1, 2} form the $MaxIS$ (compare with corresponding disjoint separators 1 and 2).

intersect each other. Graph G is tripartite because parallel separators (along axes x, y and z) are always disjoint. Example of a graph construction is shown in Fig. 7. The maximum independent set of vertices $MaxIS$ gives us the largest set of disjoint separators of weight α. We split the object by these separators and proceed to the next iteration. Note that the set of the separators may not be unique because the graph may contain more than one $MaxIS$ set of the same cardinality. In such a case we chose the separator set randomly.

We repeat the iterations until all concave edges have been eliminated. Depending on the particular choice of w_s, α may not monotonically decrease during the iteration process. At the end, the object has been decomposed into rectangular blocks. The partitioning may sometimes produce adjacent blocks that share one side and therefore they can be merged into a single block. As soon as the iterations have been completed, we find and merge these adjacent blocks.

3.2. The weight function

As we already explained, the choice of the weight function w_s determines the sub-optimal approximation of the full-search technique. It should describe the significance of the separator for the decomposition. High weights should be given to separators, the early placement of which leads to a low number of blocks. At the same time, the evaluation of the weight of each separator should be fast enough. This is why we limit ourselves to two weight functions, both of which can be evaluated directly on the current level and do not require any recursive hierarchical calculations.

The first one simply counts the number of the concave edges which the separator eliminates when placed

$$w_s^{(1)} = |\{e \mid e \in E \wedge e \subset s\}|$$

In the example in Fig. 5, the highlighted separator has the weight $w_s^{(1)} = 4$ since it contains four concave edges.

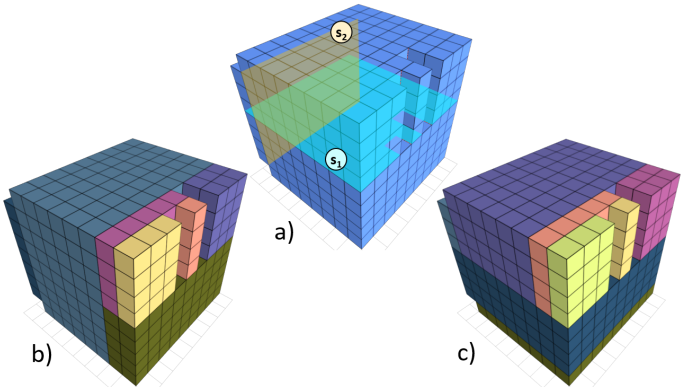


Figure 8: An example that neither $w_s^{(1)}$ nor $w_s^{(2)}$ select the optimal separators. At the first step, both weight functions prefer s_1 to s_2 because $w_{s_1}^{(1)} = 6$ and $w_{s_1}^{(2)} = 4$ while $w_{s_2}^{(1)} = w_{s_2}^{(1)} = 2$. However, placing s_1 leads to 8 blocks (see c)), while using s_2 yields the optimal decomposition into 7 blocks.

A more sophisticated choice (but also slightly more time-consuming to evaluate) which reflects the fact that the separator may also generate some new concave edges is

$$w_s^{(2)} = |\{e \mid e \subset s\}| - |\{e \mid e \perp s\}|, e \in E \tag{2}$$

which is in fact $w_s^{(1)}$ minus the number of the concave edges perpendicular to and intersected by the separator.

In the experimental section we will compare the performance of $w_s^{(1)}$ and $w_s^{(2)}$, among others.

3.3. Implementation

In the following pseudocode, we describe the algorithm more formally. Placing the separator in the object is implemented in a way that the separator becomes "final" and forms a "wall" that cannot be divided any further. We first search for all concave edges and all separators that contain them. Then we iteratively choose maximum sets of disjoint separators with the highest weight and move them to the set of walls. The concave edges, eliminated by these separators, are removed from the list and new edges (if any) are added. As

soon as the iteration process has been completed, the blocks are formed by original object surface and/or by "final inner walls" created by the separators. For each block we store the coordinates of its upper left front voxel and three block dimensions. The last step – block merging – is accomplished by lexicographic sorting the blocks w.r.t. x, y, z, identifying adjacent blocks of the same side size and updating the data in the block list. The complexity of the merging is only $\mathcal{O}(K' \log K')$ where K' is the number of blocks produced by the iterative part of the algorithm. (Note that the block merging step is due to the sub-optimality of the algorithm. If the decomposition was optimal, no merging would be possible and this step could be removed from the algorithm.)

The most time-consuming part is finding the $MaxIS$ on line 9 of the algorithm. For general graphs, this problem is NP-hard. Although the graph we work with is a tripartite one, which is much simpler than a general graph, finding the maximum independent set is still NP-hard w.r.t. the number of separators of the same weight. Theoretically, this number may be proportional to the number of all surface voxels. Actually, it is usually much lower namely for high α, but may become so high for low α that the algorithm may not be feasible. This is another substantial difference from the 2D case – finding the maximum independent set in a bipartite graph is of a polynomial complexity [21]. In the next Section, we propose an approximation of a polynomial time complexity, which we use in our implementation.

We implemented the decomposition for Node.js framework. In addition to standard node.js package written in Javascript, we created an online visual tool that runs in any modern browser that supports WebGL (such as the latest versions of Google Chrome). In this tool, the user can interactively create a custom object or upload an external object, run different decomposition methods (the proposed algorithm can be even executed step-by-step for better understanding of the process). The visualization helps in understanding the behavior of the decomposition algorithms. The tool was designed mainly for educational purposes, it is not meant for routine work. It is available online at `http://goo.gl/hAEuCg`, a sample screenshot can be seen in Fig. 9.

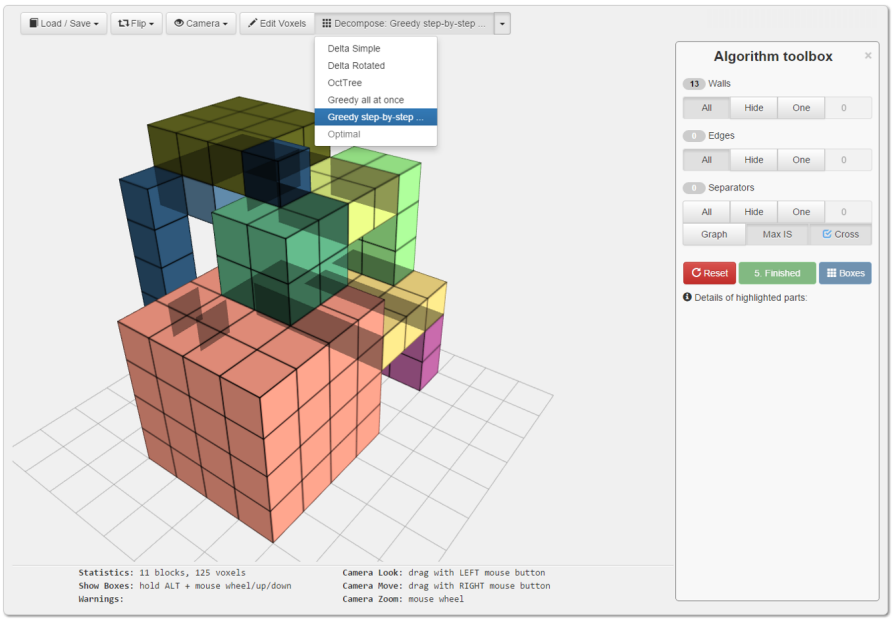


Figure 9: A sample screen shot of our online decomposition tool.

3.4. Approximating the maximum independent set

The vertices of graph $G = (V, E')$ can be clustered into three disjoint subsets P_x, P_y, P_z according to the axis that the corresponding separators are perpendicular to

$$G(V, E') = G(P_x \cup P_y \cup P_z, E'). \tag{3}$$

Clearly, each subset is composed of parallel separators which cannot intersect each other and hence G is a tripartite graph because there are no graph edges inside the individual subsets.

The complexity of finding the maximum independent set of vertices of G is exponential w.r.t. the number of the vertices and edges, which varies in individual iterations. If the number of the vertices is low, which is a typical situation at the beginning of the algorithm when the maximum weight α is

Algorithm 1 Sub-optimal 3D decomposition

$E \leftarrow$ find concave edges

$S \leftarrow$ find separators

$W \leftarrow \emptyset$ ▷ the set of final walls

while $|S| > 0$ **do**

  $w_s \leftarrow$weight(s), $\forall s \in S$ ▷ calc. weight for each sep.

  $\alpha \leftarrow \max_{s \in S}(w_s)$ ▷ calc. the max. weight

  $M \leftarrow \{s \mid w_s = \alpha \wedge s \in S\}$ ▷ separators of max. weight

  $G = (V, E') \leftarrow v_s \in V \Leftrightarrow s \in M, (v_s, v_p) \in E' \Leftrightarrow s \perp p$ ▷ create graph

  $I \leftarrow MaxIS(G)$ ▷ find max. indep. set of vertices

  $F \leftarrow \{s \mid v_s \in I\}$ ▷ seps. chosen in MaxIS become final

  $W \leftarrow W \cup F$ ▷ move final seps. to the set of walls

  $C \leftarrow \{c \mid c \perp s \wedge c \in S \wedge s \in F\}$ ▷ intersecting separators

  $N \leftarrow$ new divided separators that replace C

  $S \leftarrow (S \cap C \cap F) \cup N$ ▷ remove final seps., add divided seps.

  divide all $e \in E$ that intersect any $s \in F$ ▷ split edges that inters. walls

end while

convert voxels bounded by walls $w \in W$ into rectangular blocks

merge adjacent blocks

high, the exponential time may be acceptable. Finding the $MaxIS$ is equivalent to finding the maximal clique in the complement graph[7], so we adopted the popular Bron-Kerbosch algorithm [36] for the clique problem to find the $MaxIS$. This is, however, not feasible for large graphs, typically arriving in case of complex objects with many concave edges at the iteration levels when the weight approaches one.

Very simple approximation for $MaxIS$ is just to take the largest subset among P_x, P_y, P_z instead (let us denote it as $IS^{(1)}$). The independence is guaranteed but for most (especially large) graphs this approximation is far from the actual $MaxIS$.

A better approach is to treat this problem in tripartite graphs as an extension of the bipartite graph problem. We choose two largest vertex subsets among P_x, P_y, P_z and consider a subgraph of G (let us denote it as G_2), which is a bipartite graph. On G_2 we find the exact maximum independent set $I_2 = MaxIS(G_2)$. This is solvable in a polynomial time thanks to the König's theorem [37]. We implemented this step by means of the maximum network flow algorithm by Edmonds and Karp [35] of a time-complexity $\mathcal{O}(VE'^2)$ and also alternatively by the Dinic's algorithm [38] with the complexity $\mathcal{O}(V^2E')$. Our algorithm selects automatically the method which is more efficient for the particular graph. Finally, we unify I_2 with those vertices from the remaining third part of the graph which are not adjacent to any vertex of the selected independent set. We denote this final independent set as $IS^{(2)}$ (see Fig. 10).

The choice of how to calculate/approximate the $MaxIS$ can be in our implementation done by the user. It is always a trade-off between the time efficiency and the size of the independent set (which consequently influences the number of blocks). The optimal solution provides the correct maximum set, but it is NP-hard and thus for complicated objects it can run unacceptably long time.

[7]Complement graph H to the given graph G consists of the same set of vertices and complementary set of edges, i.e. two distinct vertices of H are adjacent if and only if they are not adjacent in G.

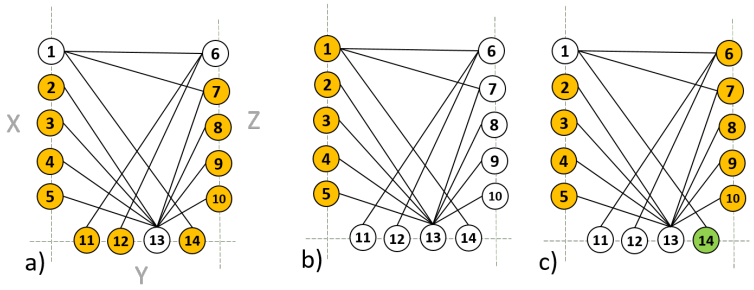


Figure 10: Finding maximum independent set in a tripartite graph. (a) optimal solution $MaxIS$, (b) an approximative heuristic $IS^{(1)}$ where only the largest part is taken, (c) better approximative heuristic $IS^{(2)}$ as an extension of bipartite subgraph $MaxIS$ completed with the vertex No. 14 from the third part of the graph.

$IS^{(1)}$ is retrieved very quickly, but the set is much smaller and thus leads to more blocks in the final decomposition. $IS^{(2)}$ provides a very good compromise, as demonstrated in the next Section by experiments.

4. Experiments

The main goal of this Section is to compare the proposed decomposition method with the state of the art Octree algorithm [39], the BSP algorithm where the separator was chosen such that it minimizes the number of voxels it passes through, and namely with the generalized delta method, which is known from 2D to be very powerful. The second goal is to study the performance of various modifications of the proposed algorithm. We verify that the polynomial heuristic functions $IS^{(1)}$ and $IS^{(2)}$ provide good approximation of the optimal NP-hard solution of the $MaxIS$. Additionally, we compare the two different weight functions that evaluate the separators' significance and verify that the enhanced $w^{(2)}$ performs significantly better than $w^{(1)}$.

Table 1: The number of the blocks for the entire DB achieved by various methods.

Method	Graph IS method	Weight function	Total No. of blocks $[10^3]$	Mean time per object [s]
Octree	N/A	N/A	3413	1.3
BSP	N/A	N/A	512	4.6
3GDM	N/A	N/A	458	0.7
Proposed	$MaxIS$	$w^{(1)}$	448	88.6
Proposed	$MaxIS$	$w^{(2)}$	445	107.5
Proposed	$IS^{(1)}$	$w^{(2)}$	450	67.4
Proposed	$IS^{(2)}$	$w^{(2)}$	445	83.3

4.1. Models from the McGill database

The first round of experiments was run on a database of 416 voxelized models from the McGill 3D Shape Benchmark[8] [23] (we denote this set as "MDB"). All models have been inscribed into a $128 \times 128 \times 128$ cube. Each object has a different volume, but together the whole MDB contains more than 13.5 millions of voxels.

We decomposed all shapes by the OTD, BSP, 3GDM, and by the proposed method with various settings (see Fig. 11 for some examples). The test results are summarized in Table 1. In the fourth and fifth rows, we used $MaxIS$ algorithm with the separator weights $w^{(1)}$ and $w^{(2)}$, respectively. Comparison of the performance of these two weights was done by Wilcoxon test. The null hypothesis was that there is no significant difference between these two sample decompositions. The null hypothesis was rejected with p-value < 0.001, which led us to the conclusion that $w^{(2)}$ performs significantly better. On the last two rows of the table we can see the most important results of the experiment - decomposition achieved by heuristics $IS^{(1)}$ and $IS^{(2)}$. In both cases, solely the

[8]Some of these objects can be found in the Princeton Shape Benchmark [22], too.

Figure 11: Example of the models from the McGill database [23] and their decomposition by the proposed method.

better weight $w^{(2)}$ was used. A surprising result is that the polynomial heuristic $IS^{(2)}$ yields almost the same number of blocks as the optimal NP-hard algorithm $MaxIS$. This was confirmed by the Wilcoxon test – the null hypothesis was accepted with the p-value > 0.1. This result proves the efficiency of $IS^{(2)}$ algorithm. When applying $IS^{(1)}$ heuristic, the decomposition works faster but in average it yields slightly higher number of blocks. Since the differences are more or less consistently spread over the whole database, the Wilcoxon test rejected the null hypothesis with p-value < 0.001. A comparison with the 3GDM is clearly in favor of the proposed method. For both $IS^{(1)}$ and $IS^{(2)}$ (and of course also for $MaxIS$), the Wilcoxon test confirmed that 3GDM performed significantly worse on this database.

Summarizing, the most important result of the test is the following. Polynomial heuristics $IS^{(2)}$ with the weight $w^{(2)}$ is statistically equivalent (in terms of the block number) to NP-hard $MaxIS$ algorithm and is at the same time significantly better than all other tested methods. This is expressed visually in Fig. 12.

4.2. Random cubes

The aim of this experiment was to compare the performance of the 3GDM and the proposed method on the set of irregular, less compact, objects. We used one hundred $32 \times 32 \times 32$ cubes inside which we randomly "carved" holes of an average density 50%. The proposed method was applied in both configurations, i.e. using $MaxIS$ and $IS^{(2)}$ algorithms. A sample cube along with its decompositions can be seen in Fig. 13. The individual number of blocks are shown "cube by cube" in Fig. 14 and the summary results are in Table 2. Similarly as on the MDB objects, we can clearly see the insignificant difference between $MaxIS$ and $IS^{(2)}$, and significantly worse performing 3GDM and BSP (these conclusions were again confirmed by the Wilcoxon test).

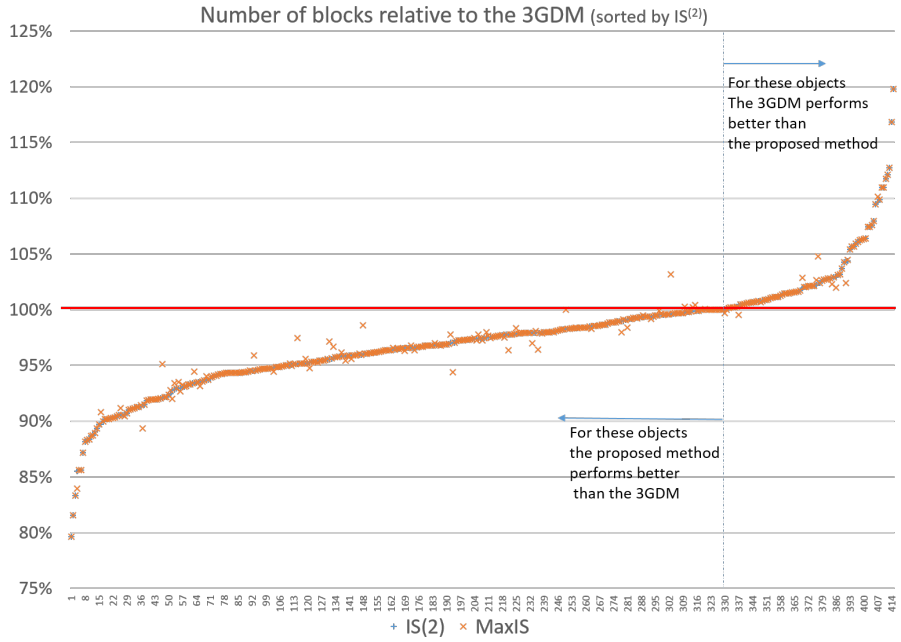
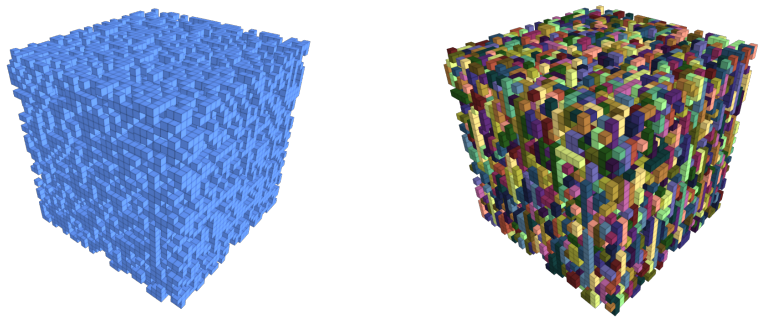


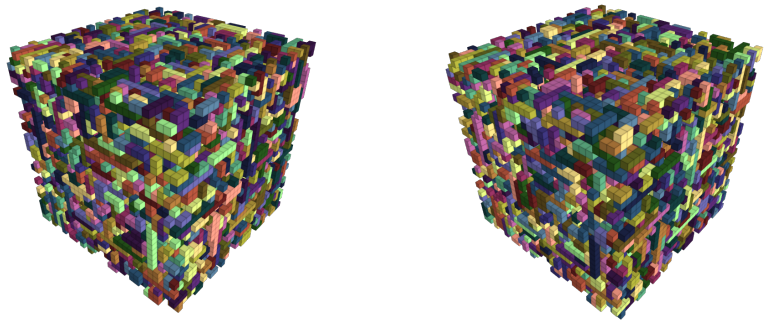
Figure 12: The differences of the numbers of blocks between the 3GDM and the proposed method. Red solid line corresponds to the 3GDM which is taken as an etalon. The objects have been sorted according to the differences between 3GDM and $IS^{(2)}$ (denoted as blue "+"). The differences between 3GDM and $MaxIS$ are denoted as orange "×".

Table 2: The number of the blocks for the set of 100 random cubes by various methods.

Method	Total No. of blocks $[10^3]$
BSP	838
3GDM	717
$MaxIS$	648
$IS^{(2)}$	654



(a) (b)



(c) (d)

Figure 13: A sample random cube (16449 voxels) (a), its 3GDM decomposition (7174 blocks) (b), $MaxIS$ decomposition (6458 blocks) (c), and $IS^{(2)}$ decomposition (6521 blocks) (d).

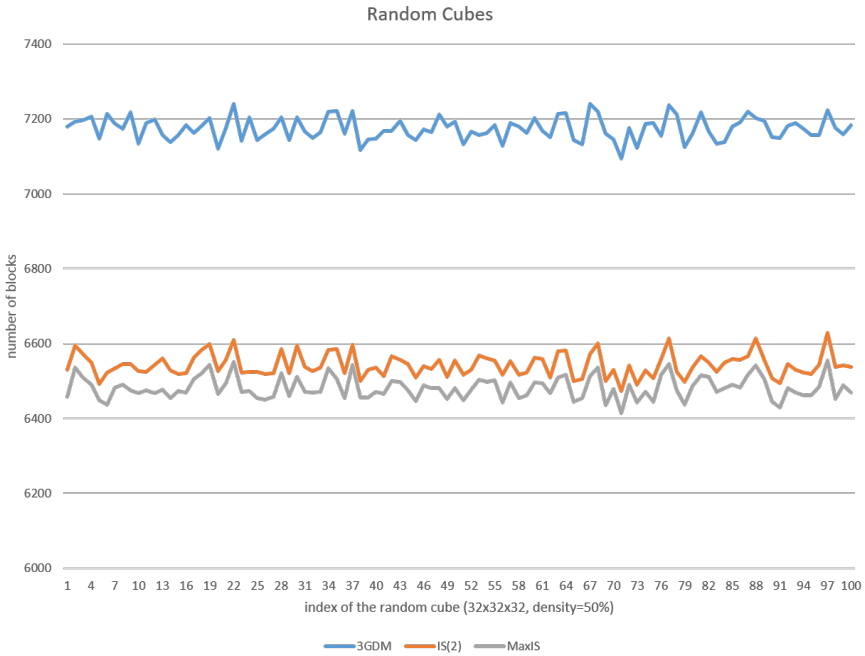


Figure 14: The number of blocks in the decomposition of 100 random cubes. Blue line – 3GDM, orange line – $IS^{(2)}$, gray line – $MaxIS$.

5. Applications

Potential applications of the proposed decomposition method can be found in all areas where decomposition of 3D shapes is required and where the number of the blocks is the main issue. This is typically if the decomposition is performed off-line, if it is then used many times in subsequent calculations, and if the number of blocks influences substantially the time and/or the cost of a subsequent processing. If, on the other hand, realtime or close-to-realtime decomposition is required as a primary criterion, then the 3GDM method provides the best solution.

5.1. Compression

Our method can be used in 3D shape encoding/compression, both loss-less and lossy ones. In a loss-less compression, we store the position and the size of each block. To optimize the compression ratio, we order the blocks according to their size such that the blocks of the same size form a substring. Then we store only the positions of the blocks while the size is stored only once for each substring. In a lossy compression, we throw away the smallest blocks, typically from $1\times1\times1$ to a certain limit. This significantly improves the compression ratio but many shape details may disappear when the object has been reconstructed.

5.2. Feature calculation

Many features, which have been proposed for 3D shape description and recognition, are of the form of an integral transformation

$$M_{\mathbf{p}}^{(f)} = \int_{\Omega} \pi_{\mathbf{p}}(\mathbf{x}) f(\mathbf{x}) \mathrm{d}\mathbf{x} \ . \tag{4}$$

where $\mathbf{p}$ is a 3D multi-index, $\{\pi_{\mathbf{p}}(\mathbf{x})\}$ is a set of basis functions of the image space (transformation kernels), $f(\mathbf{x})$ is characteristic function of the shape, and Ω is a bounded subset of R^3. Fourier coefficients, wavelet coefficients, and image moments are few popular examples [40]. If we decompose the object into

K disjoint blocks B_k, Eq. (4) can be rewritten as

$$M_{\mathbf{p}}^{(f)} = \sum_{k=1}^{K} \int_{B_k} \pi_{\mathbf{p}}(\mathbf{x}) \mathrm{d}\mathbf{x} \ . \tag{5}$$

If the basis functions $\pi_{\mathbf{p}}(\mathbf{x})$ can be integrated on a rectangular region by means of primitive functions and Newton-Leibnitz theorem in $\mathcal{O}(1)$ time (which is the case of all polynomial and harmonic bases), then the evaluation of $M_{\mathbf{p}}^{(f)}$ from Eq. (5) is of $\mathcal{O}(K)$ complexity while the direct calculation from Eq. (4) is proportional to the total number of the object voxels.

The object features are typically computed for a large set of the basis functions and used repeatedly, so the time benefit of the decomposition may be really huge even if the decomposition itself might be relatively slow.

5.3. Fast convolution

When we calculate a convolution of 3D image (graylevel or color) f with a binary kernel h, we can benefit from the decomposition as well. If we decompose the support of h into disjoint blocks, then we have

$$f * h = f * \sum_{k=1}^{K} h_k = \sum_{k=1}^{K} f * h_k, \tag{6}$$

where h_k is a characteristic function of block B_k.

The evaluation of convolution of arbitrary f with rectangular B_k in a single voxel can be accomplished in $\mathcal{O}(1)$ time regardless of the block size. We precompute partial sums of f in all three dimensions, so we create an auxiliary array g such that

$$g(L, M, N) = \sum_{\ell=1}^{L} \sum_{m=1}^{M} \sum_{n=1}^{N} f(\ell, m, n).$$

This array is precomputed only once for the given f and may be used for any h. The convolution $f * h_k$ in a certain voxel is then evaluated just from the values of g which correspond to the corners of properly shifted block B_k. This approach is significantly faster than calculating convolution from the definition, which is proportional to the kernel size, and also than using FFT; especially

when multiple convolutions with the same kernel are to be calculated. Such a situation arises for instance in deblurring/sharpening of CT images (and other data of a similar nature) when the blurring kernel is known since it characterizes the imaging device. Iterative deconvolution methods, such as popular Richardson-Lucy algorithm, iteratively estimate the input image, evaluate many times convolutions with the same kernel, and check in each iteration the similarity to the blurred acquired image.

The kernel decomposition is particularly efficient if the kernel matrix has a full or close-to-full rank because in that case we cannot effectively use any fast convolution algorithm based on kernel factorization.

5.4. Manufacturing

Manufacturing of 3D structures is often done by assembling them from simple components. If these components are rectangular blocks, then our algorithm can be advantageously applied because the production cost and time are proportional to the number of blocks, while durability of the product uses to be inversely proportional to it. The time of decomposition, which is performed on a computer model of the product, is negligible comparing to the total production time.

6. Concluding discussion

In this paper, we presented an original method of block-wise decomposition in 3D. The method is a double approximation of the optimal algorithm, which is NP-complete and practically infeasible. We proposed the criterion for the separator selection in the first heuristic approximation. In the second approximation, the maximum independent set in a tripartite graph, the finding of which is again NP-complete (and to our knowledge is also hard to approximate), is replaced by a polynomial sub-optimal solution. We proved by large-scale experiments that the proposed method is statistically better than the 3GDM, which is the best one among the existing methods in terms of the number of blocks. This

determines that potential applications of the proposed method can be found namely in the tasks where it is more desirable to keep the number of blocks as low as possible rather than to minimize the decomposition runtime.

It is interesting that even though the proposed algorithm is generally suboptimal, it is optimal on specific shapes. A trivial example of such class are objects, which do not have any cogrid concave edges. Another class where our method performs optimally, is formed by objects all concave edges of which are parallel.[9] This is not true for the main competitor, 3GDM, which is still suboptimal on these objects.

Although in the paper we have been dealing with binary shapes only, all methods can be theoretically used for graylevel and color 3D images as well. A graylevel image can be expressed as a sum of several binary 3D images multiplied by a proper constant, which can be obtained either as intensity slices or bit slices (these techniques were proposed for 2D images in [41], [42] and can be adapted to 3D case readily). We can decompose each slice independently by means of our algorithm. However, these binary "images" use to be highly fragmented (especially low bit planes resemble a "random chessboard") and their decomposition typically yields high number of blocks. Hence, this kind of decomposition is of a little practical importance for graylevel/color 3D images.

For the sake of completeness, we should mention that the format of binary images used in some databases is not based on a voxel representation. A common representation of a shape is by triangular patches (facets) of the surface. In that case, only the vertices of the triangles are stored. Such a representation may be often used directly for compression, feature calculation, and other tasks. This is, however, not true in general. Calculating convolution and in-

[9] This property can be proven formally. First, we prove that in this case all concave edges must have their beginning in the same object face. The same is true for the edge ends. This means that such object is an "elongation" of a 2D shape, which forms the object face, into the third dimension. Our algorithm is equivalent to 2D optimal decomposition of the base and "elongation" of the rectangles into blocks. Hence, our method must be optimal, too.

tegral transformations in triangular representation may be very tricky or even impossible. If we want to find a rectangular decomposition, we have to convert the triangular representation to the voxel one first. Although several conversion algorithms exist (see, for instance, [40] for one of the simplest methods), this approach is not very efficient and the triangular representation is mostly used without any conversion to voxels.

References

[1] F. B. Neal, J. C. Russ, Measuring Shape, CRC Pres, 2012.

[2] S. Marchand-Maillet, Y. M. Sharaiha, Binary Digital Image Processing: A Discrete Approach, Academic Press, 1999.

[3] J. M. Keil, Polygon decomposition, in: Handbook of Computational Geometry, Elsevier, 2000, pp. 491–518.

[4] M. F. Zakaria, L. J. Vroomen, P. Zsombor-Murray, J. M. van Kessel, Fast algorithm for the computation of moment invariants, Pattern Recognition 20 (6) (1987) 639–643.

[5] M. Dai, P. Baylou, M. Najim, An efficient algorithm for computation of shape moments from run-length codes or chain codes, Pattern Recognition 25 (10) (1992) 1119–1128.

[6] B. C. Li, A new computation of geometric moments, Pattern Recognition 26 (1) (1993) 109–113.

[7] I. M. Spiliotis, B. G. Mertzios, Real-time computation of two-dimensional moments on binary images using image block representation, IEEE Transactions on Image Processing 7 (11) (1998) 1609–1615.

[8] J. Flusser, Refined moment calculation using image block representation, IEEE Transactions on Image Processing 9 (11) (2000) 1977–1978.

[9] C.-H. Wu, S.-J. Horng, P.-Z. Lee, A new computation of shape moments via quadtree decomposition, Pattern Recognition 34 (7) (2001) 1319–1330.

[10] J. H. Sossa-Azuela, C. Yáñez-Márquez, J. L. Díaz de León Santiago, Computing geometric moments using morphological erosions, Pattern Recognition 34 (2) (2001) 271–276.

[11] T. Suk, J. Flusser, Refined morphological methods of moment computation, in: 20th International Conference on Pattern Recognition ICPR'10, IEEE Computer Society, 2010, pp. 966–970.

[12] P. Campisi, K. Egiazarian, Blind image deconvolution: theory and applications, CRC, 2007.

[13] A. Levin, R. Fergus, F. Durand, W. T. Freeman, Image and depth from a conventional camera with a coded aperture, in: Special Interest Group on Computer Graphics and Interactive Techniques Conference SIGGRAPH'07, ACM, New York, NY, USA, 2007.

[14] W. Liou, J. Tan, R. Lee, Minimum partitioning simple rectilinear polygons in O(n log log n)-time, in: Proceeding of the fifth Annual ACM symposium on Computational Geometry SoCG'89, ACM, New York, NY, USA, 1989, pp. 344–353.

[15] K. D. Gourley, D. M. Green, A polygon-to-rectangle conversion algorithm, IEEE Computer Graphics and Applications 3 (1) (1983) 31–36.

[16] W. Lipski Jr., E. Lodi, F. Luccio, C. Mugnai, L. Pagli, On two-dimensional data organization II, in: Fundamenta Informaticae, Vol. II of Annales Societatis Mathematicae Polonae, Series IV, 1979, pp. 245–260.

[17] T. Ohtsuki, Minimum dissection of rectilinear regions, in: Proceedings of the IEEE International Conference on Circuits and Systems ISCAS'82, IEEE, 1982, pp. 1210–1213.

[18] L. Ferrari, P. V. Sankar, J. Sklansky, Minimal rectangular partitions of digitized blobs, Computer Vision, Graphics, and Image Processing 28 (1) (1984) 58–71.

[19] H. Imai, T. Asano, Efficient algorithms for geometric graph search problems, SIAM Journal on Computing 15 (2) (1986) 478–494.

[20] D. Eppstein, Graph-theoretic solutions to computational geometry problems, in: 35th International Workshop on Graph-Theoretic Concepts in Computer Science WG'09, Vol. LNCS 5911, Springer, 2009, pp. 1–16.

[21] T. Suk, C. Höschl IV, J. Flusser, Decomposition of binary images – a survey and comparison, Pattern Recognition 45 (12) (2012) 4279–4291.

[22] P. Shilane, P. Min, M. Kazhdan, T. Funkhouser, The princeton shape benchmark (June 2004).
URL http://shape.cs.princeton.edu/benchmark/

[23] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, S. Dickinson, Retrieving articulated 3-d models using medial surfaces, Mach. Vision Appl. 19 (4) (2008) 261–275. doi:10.1007/s00138-007-0097-8.

[24] Z. Ren, J. Yuan, W. Liu, Minimum near-convex shape decomposition, IEEE Trans. on on Pattern Analysis and Machine Intelligence 35 (2013) 2546–2552.

[25] I. Sivignon, D. Coeurjolly, Minimum decomposition of a digital surface into digital plane segments is NP-hard, Discrete Applied Mathematics 157 (3) (2009) 558–570. doi:10.1016/j.dam.2008.05.028.

[26] Y. Zhou, K. Yin, H. Huang, H. Zhang, M. Gong, D. Cohen-Or, Generalized cylinder decomposition, ACM Trans. Graph. 34 (6) (2015) 171:1–171:14. doi:10.1145/2816795.2818074.

[27] A. Jain, T. Thormählen, T. Ritschel, H.-P. Seidel, Exploring shape variations by 3d-model decomposition and part-based recombination, Computer

Graphics Forum 31 (2pt3) (2012) 631–640. doi:10.1111/j.1467-8659.2012.03042.x.

[28] V. J. Dielissen, A. Kaldewaij, Rectangular partition is polynomial in two dimensions but np-complete in three, Information Processing Letters 38 (1) (1991) 1 – 6.

[29] E. Kawaguchi, T. Endo, On a method of binary-picture representation and its application to data compression, IEEE Transactions on Pattern Analysis and Machine Intelligence 2 (1) (1980) 27–35.

[30] J. Flusser, An adaptive method for image registration, Pattern Recognition 25 (1) (1992) 45–54.

[31] M. d. Berg, O. Cheong, M. v. Kreveld, M. Overmars, Computational Geometry: Algorithms and Applications, 3rd Edition, Springer-Verlag TELOS, Santa Clara, CA, USA, 2008.

[32] T. M. Murali, P. K. Agarwal, J. S. Vitter, Constructing binary space partitions for orthogonal rectabgles in practice, in: Proceedings of the 6th Annual European Symposium on Algorithms, ESA '98, Springer-Verlag, London, UK, UK, 1998, pp. 211–222.
URL http://dl.acm.org/citation.cfm?id=647908.739984

[33] A. V. Goldberg, S. Rao, Beyond the flow decomposition barrier, Journal of the Association for Computing Machinery 45 (5) (1998) 783–797.

[34] A. V. Goldberg, R. E. Tarjan, A new approach to the maximum-flow problem, Journal of the Association for Computing Machinery 35 (4) (1988) 921–940.

[35] J. Edmonds, R. M. Karp, Theoretical improvements in algorithmic efficiency for network flow problems, Journal of the Association for Computing Machinery 19 (2) (1972) 248–264.

[36] C. Bron, J. Kerbosch, Algorithm 457: Finding all cliques of an undirected graph, Commun. ACM 16 (9) (1973) 575–577. doi:10.1145/362342.362367.

[37] W. Cook, W. Cunningham, W. Pulleyblank, A. Schrijver, Combinatorial Optimization, Wiley Series in Discrete Mathematics and Optimization, Wiley, 2011.

[38] E. A. Dinic, Algorithm for solution of a problem of maximum flow in a network with power estimation, Soviet Math Doklady 11 (1970) 1277–1280.

[39] D. Meagher, Octree encoding: A new technique for the representation, manipulation and display of arbitrary 3-d objects by computer, Tech. Rep. IPL-TR-80-111, Rensselaer Polytechnic Institute (October 1980).

[40] J. Flusser, T. Suk, B. Zitová, 2D and 3D Image Analysis by Moments, Wiley, 2016, in print, to appear in 2016.

[41] G. A. Papakostas, E. G. Karakasis, D. E. Koulouriotis, Efficient and accurate computation of geometric moments on gray–scale images, Pattern Recognition 41 (6) (2008) 1895–1904.

[42] I. M. Spiliotis, Y. S. Boutalis, Parameterized real-time moment computation on gray images using block techniques, Journal of Real-Time Image Processing 6 (2) (2011) 81–91.

Recognition of Images Degraded by Gaussian Blur

Jan Flusser, *Senior Member, IEEE*, Sajad Farokhi, Cyril Höschl IV, Tomáš Suk, Barbara Zitová, and Matteo Pedone

***Abstract*— In this paper, we propose a new theory of invariants to Gaussian blur. We introduce a notion of a primordial image as a canonical form of all Gaussian blur-equivalent images. The primordial image is defined in spectral domain by means of projection operators. We prove that the moments of the primordial image are invariant to Gaussian blur and we derive recursive formulas for their direct computation without actually constructing the primordial image itself. We show how to extend their invariance also to image rotation. The application of these invariants is in blur-invariant image comparison and recognition. In the experimental part, we perform an exhaustive comparison with two main competitors: 1) the Zhang distance and 2) the local phase quantization.**

***Index Terms*— Blurred image, object recognition, blur invariant comparison, Gaussian blur, projection operators, image moments, moment invariants.**

I. Introduction

IMAGE recognition/classification in general is an extremely broad area which apparently cannot be resolved by a single, always-optimal method. This is why numerous specific formulations of the problem have appeared, which consequently has resulted in many approaches and particular algorithms. Some of them have already become an established discipline of image analysis while some others are still undergoing initial development. One of the representatives of the latter group are methods for recognition of images which are degraded by a uniform Gaussian blur.

Few years ago, this task was considered a borderline problem. Thanks to the rapid development of imaging sensors and technologies that are nowadays available everywhere, the challenge of recognizing Gaussian-blurred images has started to appear more and more often in practice which consequently has attracted the attention of the researchers.

The mathematical formulation of the problem is well known in image processing. Capturing an ideal scene f by an imaging device with the point-spread function (PSF) h, the observed image g is a convolution of both

$$g(x, y) = (f * h)(x, y). \tag{1}$$

This linear space-invariant image formation model, even if it is very simple, is a reasonably accurate approximation of many imaging devices and acquisition scenarios. In this paper, we concentrate our attention to the case when the PSF is a Gaussian function with unknown parameters.

Gaussian blur appears whenever the image was acquired through a turbulent medium and the acquisition/exposure time is by far longer than the period of Brownian motion of the particles in the medium. Ground-based astronomical imaging through the atmosphere, taking pictures through a fog, underwater imaging, and fluorescence microscopy are typical examples of such situation (in some cases, the blur may be coupled with a contrast decrease). Gaussian blur is also introduced into the images as the sensor blur which is due to a finite size of the sampling pulse; this effect is, however, mostly of low significance. Moreover, Gaussian kernel is often used as an approximation of some other blurs which are too complicated to work with them exactly. Gaussian blur is sometimes even introduced into the image intentionally, for instance to suppress additive noise, to "soften" the image or to perform local averaging before the image is downscaled (see Fig. 1 for some examples). Numerous examples of the Gaussian convolution can be found outside the image processing area – particle transportation, diffusion process, time-development of a heat distribution in a mass, and photon scattering in radiation physics are few examples. Most of them are represented by 2D or 3D functions which can be visualized, that brings us back to image processing. So, we can see there is actually a demand for developing the tools designed particularly for processing Gaussian-blurred images.

When we need to classify a blurred image g against a database of clear images, we have basically three options. The most time-expensive one is to generate all possible blurred versions of all templates (i.e. blurring with Gaussians the variances of which fill a reasonable, properly sampled interval) and incorporate them into the database. This brute-force approach is not practically feasible. Another approach relies on the solution of the inverse problem, when the blur is removed from the input image and the deblurred image is then classified by any standard technique. This process contains semi-blind image deconvolution (the term "semi-blind" is used because we know the parametric form of the kernel but its parameters are unknown) which is in the case of a Gaussian kernel an unstable, ill-posed problem. Unlike motion blur and out-of-focus blur, Gaussian blur does

Manuscript received April 7, 2015; revised July 29, 2015 and October 18, 2015; accepted December 14, 2015. Date of publication December 23, 2015; date of current version January 7, 2016. This work was supported by the Czech Science Foundation under Grant GA15-16928S. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Dacheng Tao.

J. Flusser, C. Höschl IV, T. Suk, and B. Zitová are with the Institute of Information Theory and Automation, Czech Academy of Sciences, Prague 182 08, Czech Republic (e-mail: flusser@utia.cas.cz; hoschl@utia.cas.cz; suk@utia.cas.cz; zitova@utia.cas.cz).

S. Farokhi is with the Institute of Information Theory and Automation, Czech Academy of Sciences, Prague 182 08, Czech Republic, and also with the Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad 85141-43131, Iran (e-mail: fsajad2@utia.cas.cz).

M. Pedone is with the Center for Machine Vision Research, Department of Computer Science and Engineering, University of Oulu, Oulu FI-90014, Finland (e-mail: matped@ee.oulu.fi).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2512108

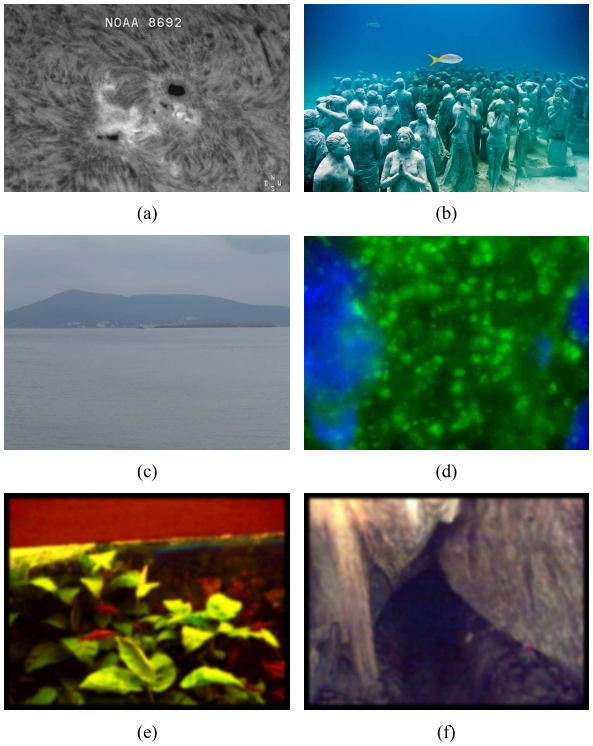


Fig. 1. Examples of the Gaussian blur: (a) the image of the sunspot blurred by atmospheric turbulence, (b) the underwater photo blurred by light dispersion, (c) a picture blurred due to a fog, (d) the image of axon boutons from wide-field epifluorescence microscopy, (e) the snap of an aquarium and (f) the snap from the cave. The last two pictures were originally very noisy because of poor light conditions, the noise was suppressed by applying a computer-generated Gaussian blur.

not introduce any zero patterns into the spectrum of the image, which are in the other cases employed for parameter estimation. Another difficulty is that the Gaussian can be factorized into an arbitrary (theoretically infinite) number of convolution terms, each of them being again a Gaussian. Hence, deconvolution algorithms cannot in principle remove Gaussian blur if no prior information is available. If the blur size (i.e. the variance of the Gaussian) was known, then we could apply a convolution with an inverse kernel (which can be synthesized by Hermite polynomials) or Wiener deconvolution in the frequency domain to deblur the image. Unfortunately, in image processing this scenario is not realistic because the blur size uses to be unknown, which makes the deblurring difficult. Only few semi-blind deconvolution methods w.r.t. Gaussian blur have been published. They first try to estimate the size (variance) of the blur and perform a non-blind deconvolution. Honarvar *et al.* [1] and Honarvar and Flusser [2] proposed to perform the deconvolution in the moment domain but his algorithm contains a time-consuming search in the parametric space and is sensitive to overestimation of the Gaussian variance. The APEX method [3] estimated the blur variance by fitting the image spectrum in the Fourier domain. There exist also several local methods that estimate the blur size by investigating the response on a point source or on an ideal edge [4], [5]. A common weakness of these methods is their sensitivity to noise and the necessity of the prior knowledge where an ideal point or edge is located. Xue and Blu [6] proposed to estimate the blur variance by minimizing a proper functional and then to apply a non-blind Wiener filtering. As in the previous cases, the method is sensitive to the variance overestimation and relatively time-consuming.

The third and the most promising approach is based on the idea that for blur-insensitive recognition we do not need to restore the query image. We only need to have its representation (possibly low-dimensional and lossy) which is robust w.r.t. Gaussian blur. We are looking for a blur-invariant image descriptor I, which is a functional defined on the space of all images, such that

$$I(f) = I(f * h) \tag{2}$$

for any Gaussian kernel h. The existence of such Gaussian blur invariants is theoretically possible thanks to the fact that the Gaussian convolution is closed under the composition.[1] The closure property is an essential necessary condition. Imagine a set S of functions (convolution kernels) which would not be closed under convolution. Then $I(f) = I(f * h_1) = I(f * h_1 * h_2)$ for arbitrary $h_1, h_2 \in S$ but obviously $(h_1 * h_2)$ may lie outside S. So, the functional I must be invariant to a convolution with a broader set of kernels. Such set is called *convolution closure* of S and we denote it as $C(S)$. If $S \neq C(S)$, then looking for the specific blur invariants w.r.t. S does not make sense. All such invariants must be at the same time invariant w.r.t. $C(S)$.

The idea of designing blur invariant functionals appeared about 20 years ago in the papers by Flusser *et al.* [7] and Flusser and Suk [8]. They proposed a system of blur invariants which are recursive functions of standard (geometric) moments of the image and proved their invariance under a convolution with arbitrary *centrosymmetric* kernel. These invariants, along with the centrosymmetry assumption, have been adopted by numerous researchers. They have become very popular image descriptors and have found a number of applications, namely in matching and registration of satellite and aerial images [8]–[12], in medical imaging [13]–[15], in normalizing blurred images into canonical forms [16], [17], in blurred digit and character recognition [18], in robot control [19], in image forgery detection [20], [21], in traffic sign recognition [22], [23], in fish shape-based classification [24], in wood industry [25], [26], and in cell recognition [27].

Several authors have further developed the theory of blur invariants. Combined invariants to convolution and to rotation were introduced by Flusser and Zitová [28], who also reported their successful usage in satellite image registration [29] and in camera motion estimation [30]. Combined invariants

[1]The set of all Gaussian functions with the binary operation convolution is a commutative monoid, i.e. a semigroup with a unit element. The closure property holds also for point-wise multiplication, so the Gaussians form a commutative ring. This assertion is valid for the set of normalized as well as unnormalized Gaussians. The Gaussian family is not the only parametric family of functions with the closure property to convolution; we recall α-stable distributions known in statistics.

both to convolution and affine transform was published by Zhang *et al.* [17] and Suk and Flusser [31]. Their use for aircraft silhouette recognition [32], for sign language recognition [33], for the classification of winged insect [34] and for robust digital watermarking [35] was reported.

Some researchers attempted derivation of blur invariants which are functions of orthogonal moments rather than of the geometric moments. Legendre moments [36]–[39], Zernike moments [40]–[42], and Chebyshev moments [43] were employed for this purpose. Zuo *et al.* [44] even combined moment blur invariants and SIFT features [45] into a single vector with weighted components but without a convincing improvement. However, as was proved by Kautsky and Flusser [46], moment invariants in any two different polynomial bases are mutually dependent and theoretically equivalent.

Some other authors constructed the blur invariants in Fourier domain. Ojansivu and Heikkilä [47], [48] and Tang *et al.* [49] used blur-invariant properties of Fourier transform phase for image registration and matching. Their idea was later significantly generalized by Pedone *et al.* [50], [51]. Popular method of the *Local phase quantization* (LPQ) [52]–[55] also belongs to this group.

In almost all papers mentioned above, the invariance property was considered–exactly as in the original paper [8]–only to centrosymmetric PSF's. Few authors were apparently aware of this limitation which decreases the discrimination power and tried to construct invariants to more specific blurs. Flusser et al. derived invariants to motion blur [56], to axially symmetric blur in case of two axes [57], to circularly symmetric blur [58], and to arbitrary N-fold symmetric blur [59].

All the above methods do not use the parametric form of the PSF at all. They can be applied to Gaussian blur as well, because the Gaussian kernel is a special case of symmetric kernels. However, these methods cannot in principle reach the maximum possible discrimination power. To understand the discrimination power of the blur-invariant methods, we have to analyze the null-space of the respective invariants. The null-space is always formed by the functions with the same symmetry as the kernel symmetry the method is invariant to. To see that, we may imagine that the object is a blur kernel applied on the delta function. This means, invariants to centrosymmetric blur cannot discriminate among all centrosymmetric objects, invariants to circularly symmetric blur are not able to discriminate the circularly symmetric objects, etc. For instance the circle and the ring blurred by a Gaussian cannot be distinguished by invariants to centrosymmetric and circularly symmetric kernels but can be distinguished by invariants to Gaussian blur. For an optimal discriminabilty we need specific invariants exactly w.r.t. the blur which is present in the image. Unfortunately, the Gaussian blur invariants cannot be easily obtained as a special case of the earlier methods (even if the idea of projection operators we employ in this paper is similar to that one we proposed in [59]).

Only few attempts to derive invariants (2) w.r.t. Gaussian blur have been reported so far. Most of them are heuristics lacking a deeper mathematical analysis. Liu and Zhang [60] realized that the complex moments of the image, one index of which is zero, are invariant to Gaussian blur. Xiao *et al.* [61] seemingly derived invariants to Gaussian blur but he did not employ the parametric Gaussian form explicitly. He only used the circular symmetry property which led to an incomplete invariant system. Gopalan *et al.* [62] derived another invariant set without assuming the knowledge of the parametric shape of the kernel but imposed a limitation of its support size. Flusser et al. mentioned an idea of Gaussian blur invariants in [63] without presenting the details and without testing their applicability.

An interesting approach, one of very few which have been proposed specifically for Gaussian blur and which works with a parametric form of the PSF, was proposed by Zhang *et al.* [64], [65]. They derived a blur-invariant distance measure d between two images which fulfills the condition

$$d(f_1, f_2) = d(f_1 * h, f_2) \tag{3}$$

for any Gaussian kernel h. Although the blur invariants are not explicitly defined, the invariant distance measure (3) can be used for object classification in a similar manner. The authors reported its good performance. The paper [65] published in this Transactions motivated us to perform a detailed study of their method, to analyze its pros and cons, and to propose a different approach based on invariants of the type (2) which outperforms the Zhang's method in several aspects.

The paper is organized as follows. Section II recalls the Zhang's method [65]. The new invariants based on projection operators are introduced in Section III. Section IV presents an experimental comparison of these two competing approaches along with a comparison to two general-purpose methods–cross correlation and LPQ.

II. Recalling the Zhang's Method

The main idea of the method proposed in [65] is simple and elegant, even if some steps are hidden behind relatively complicated mathematical formalism which employs Riemannian manifolds. Here we briefly summarize the Zhang's method in a more transparent way.

The blur model used in [65] is supposed to be as in Eq. (1), where h is assumed to be a 2D circularly symmetric centralized Gaussian function $G_\sigma(x, y)$ which is defined as

$$G_\sigma(x, y) = G_\sigma(x) G_\sigma(y), \tag{4}$$

where $\sigma > 0$ and $G_\sigma(x), G_\sigma(y)$ are 1D Gaussian functions of a traditional shape[2]

$$G_\sigma(x) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

and of the same variance σ^2.

Let f_1, f_2 be two images to be compared (their content as well as their blur level are generally different). First, the blur level of each of them is estimated by a proper blur measure. The authors used the integral of the image Laplacian but in principle any of the popular blur measures reviewed in [66] can be employed in this step. Both images are then brought

[2]Let us extend this definition by setting $G_0(x) = \delta(x)$.

to the same level of blurring which is chosen as the blur level of the more blurred image. This means that the (assumably) sharper image is blurred by a Gaussian kernel of a proper size to reach the same level of blur as the other image. This step should ensure that the distance measure becomes independent of the image blurring. Then the distance $d(f_1, f_2)$ is defined as a geodesic distance on the surface of the ellipsoid which contains the images of the same blur level. This distance is calculated by means of an iterative "path straightening" algorithm. The only difference from a pure L_2 norm is that the distance is measured along a curve on the ellipsoid surface but still it is based on a pixel-wise comparison of the images. It should be noted that in the earlier paper by the same authors [64], a simpler weighted L_2 distance was used instead.

III. Gaussian Blur Invariants Based on Projection Operators

In this section we present an approach based on the *invariant descriptors* of the type (2). The basic conceptual difference from the Zhang's method is that these invariants are defined for a single image, while the Zhang's distance always requires a pair of images. So, we can calculate the invariant representations of the database objects/templates only once and store them in the database along with the object images. It leads to much faster recognition, as will be demonstrated practically in Section IV, and also yields a possibility of broader generalization.

The invariants are derived by means of projection operators in Fourier domain, as we will see in Theorem 1. For practical application, these complete invariants are replaced with equivalent image domain invariants, which are based on image moments. Derivation of both is the subject of the rest of this section.

A. Projection Operator in 1D

The new invariants are based on the projection of the image onto a space of unnormalized Gaussian functions, which preserves the image moments of the zeroth, the first, and the second orders. The separability of a 2D Gaussian function allows us to create a 1D theory (which is more transparent and easy to explain) first and then to generalize it to the 2D (or even N-D) case.

Let us consider a 1D "image" f, $f(x) \geq 0$, with a finite non-zero integral and finite central moments of all orders. The projection operator P_G is defined as

$$P_G(f)(x) = m_0 G_s(x) \equiv \frac{m_0}{\sqrt{2\pi}\, s} e^{-\frac{x^2}{2s^2}}, \tag{5}$$

where

$$s^2 = m_2/m_0$$

and

$$m_p = \int (x-c)^p f(x) dx \tag{6}$$

is the p-th central moment of f (with c being the centroid of f). Hence, P_G assigns each f to a centralized Gaussian multiplied by m_0 such that the central moments up to the second order of f and $P_G(f)$ are equal. In other words, $P_G(f)$ is the "closest" unnormalized Gaussian to f in terms of the first three moment values. In this sense, P_G can be considered a projector onto the set of unnormalized Gaussian functions.[3]

The operator P_G exhibits several interesting properties.

- Operator P_G is idempotent, i.e. $P_G(P_G(f)) = P_G(f)$.
- Operator P_G is multiplicative, i.e. $P_G(af) = aP_G(f)$ for any constant $a > 0$.
- If f is an (unnormalized) Gaussian, then $P_G(f) = f$ and vice versa.
- Any function f can be expressed as $f = P_G(f) + f_n$, where f_n can be considered a "non-Gaussian" part of f.
- The equality $P_G(f_1) = P_G(f_2)$ defines an equivalence relation on the image space. The classes of equivalence are formed by the functions of the same zeroth and second central moments.

An important property of P_G, which will be later used for construction of the invariants, is its relationship to a convolution with a Gaussian kernel. It holds, for any f and σ,

$$P_G(f * G_\sigma) = P_G(f) * G_\sigma. \tag{7}$$

To see this, we have to establish the relation between the second-order moments $m_2^{(g)}$ on one hand and $m_2^{(f)}, m_2^{(h)}$ on the other hand. We recall (see [63] for details) that in general, for arbitrary f, h, and p, the moments are transformed under a convolution as

$$m_p^{(g)} = \sum_{k=0}^{p} \binom{p}{k} m_k^{(h)} m_{p-k}^{(f)}. \tag{8}$$

If $h(x) = G_\sigma(x)$, its moments are

$$m_k^{(h)} = \sigma^k (k-1)!! \tag{9}$$

for any even k. The symbol $k!!$ means a double factorial, $k!! = 1 \cdot 3 \cdot 5 \cdots k$ for odd k, and by definition $(-1)!! = 0!! = 1$. For any odd k the moment $m_k^{(h)} = 0$ due to the symmetry of the Gaussian function. Hence, (8) obtains the form

$$m_p^{(g)} = \sum_{\substack{k=0 \\ k \text{ even}}}^{p} \binom{p}{k} \sigma^k (k-1)!! \cdot m_{p-k}^{(f)}. \tag{10}$$

In particular,

$$m_0^{(g)} = m_0^{(f)}$$

and

$$m_2^{(g)} = m_2^{(f)} + \sigma^2 m_0^{(f)}.$$

Now we can see that

$$P_G(f * G_\sigma)(x) = m_0 G_{\sqrt{(s^2+\sigma^2)}}(x) = (P_G(f) * G_\sigma)(x) \tag{11}$$

(the latter equality follows from the fact that the convolution of two Gaussians is again a Gaussian with the variance being the sum of the input variances).

[3]However, it is not a projector in the common meaning, since it is not a linear operator and the Gaussians do not form a vector space.

B. 1D Gaussian Blur Invariants in the Fourier Domain

Now we can formulate the central Theorem of this paper.

Theorem 1: Let f be an image function. Then

$$I_G(f)(u) = \frac{\mathcal{F}(f)(u)}{\mathcal{F}(P_G(f))(u)}$$

is an invariant to Gaussian blur, i.e. $I_G(f) = I_G(f * G_\sigma)$ for any blur parameter σ.

The proof follows immediately from Eq (7). Note that I_G is invariant also to the contrast stretching, $I_G(f) = I_G(af)$.

What is the meaning of these invariants? The frequency domain provides us with a good insight. $I_G(f)$ is a ratio of two Fourier transforms which may be interpreted as a deconvolution. Having an image f, we seemingly "deconvolve" it by the kernel $P_G(f)$, which is the largest possible Gaussian kernel (larger kernels cannot exist because de-blurring always monotonically decreases m_2, reaching the limit at $m_2^{(\mathcal{F}^{-1}(I_G(f)))} = 0$). We call the result of this seeming deconvolution the *primordial image*

$$f_r = \mathcal{F}^{-1}(I_G(f)).$$

Hence, $I_G(f)$ can be viewed as its Fourier transform, although f_r is not an image in a common sense because the existence of $\mathcal{F}^{-1}(I_G(f))$ is not generally guaranteed and even if f_r exists, it may contain negative values.

$I_G(f)$ can be viewed as a kind of normalization of f w.r.t. Gaussian blurring of unknown extent. The primordial image plays the role of a canonical form of f, which actually is its "maximally deconvolved" non-Gaussian part. We can see a conceptual difference from the Zhang's approach [65]. To make two images comparable, Zhang blurs them to the same level of blur, which is given by the more blurred image in the pair. We (seemingly) deblur each image separately because the canonical form is independent of the other images.

The equality $I_G(f_1) = I_G(f_2)$ decomposes the image space into classes of equivalence. Fortunately, this decomposition is exactly the same as that one induced by the following relation: two functions f_1 and f_2 are equivalent if and only if there exist $a > 0$ and $\sigma \geq 0$ such that $f_1 = f_2 * aG_\sigma$ or $f_2 = f_1 * aG_\sigma$. To prove this, let us first realize that if $I_G(f_1) = I_G(f_2)$ then obviously

$$\mathcal{F}(f_1)(u)\mathcal{F}(P_G(f_2))(u) = \mathcal{F}(f_2)(u)\mathcal{F}(P_G(f_1))(u),$$

which in the image domain means

$$f_1 * P_G(f_2) = f_2 * P_G(f_1).$$

Both $P_G(f_i)$ are (unnormalized) Gaussians. Let us denote their standard deviations as σ_1 and σ_2, respectively, so we have $P_G(f_i) = a_i G_{\sigma_i}$. Let $\sigma_1 \geq \sigma_2$. We define $\sigma^2 = \sigma_1^2 - \sigma_2^2$ and $a = a_1/a_2$. Since the convolution of any two Gaussians is again a Gaussian the variance of which is the sum of two input variances, we have

$$aG_\sigma * a_2 G_{\sigma_2} = a_1 G_{\sigma_1}.$$

From this we immediately obtain

$$f_1 = f_2 * aG_\sigma$$

which completes the proof.

This is an important observation, saying that $I_G(f)$ is a *complete* description of f up to a convolution with a Gaussian and a multiplicative contrast change. In other words, $I_G(f)$ defines an *orbit* – a set of images equivalent with f. Thanks to the completeness, I_G discriminates between the images from different orbits but obviously cannot discriminate inside an orbit. In particular, I_G cannot discriminate between two Gaussians since all Gaussians lie on the orbit the root of which is the delta function.

C. 1D Gaussian Blur Invariants in the Image Domain

In principle, we can use directly $I_G(f)$ as the invariant feature vector of the same size as f but working in the Fourier domain brings two practical difficulties. Since $I_G(f)$ is a ratio, we possibly divide by very small numbers which requires an appropriate numerical treatment. Moreover, high frequencies of $I_G(f)$ use to be sensitive to noise. This can be overcome by suppressing them by a low-pass filter, but this procedure introduces a user-defined parameter (the cut-off frequency) which should be set up with respect to the particular noise level. That is why in most cases we prefer to work directly in the image domain, where invariants equivalent to $I_G(f)$ can be constructed.

To get the link between the Fourier and image domains, we use a Taylor expansion of the harmonic functions and its term-wise integration

$$\mathcal{F}(f)(u) \equiv \int_{-\infty}^{\infty} f(x) \cdot e^{-2\pi i u x} dx = \sum_{k=0}^{\infty} \frac{(-2\pi i)^k}{k!} m_k u^k. \tag{12}$$

The above formula tells us that the moments of the image are Taylor coefficients (up to a constant factor) of its Fourier transform. Taylor expansion of $\mathcal{F}(P_G(f))$ yields

$$\mathcal{F}(P_G(f))(u) = m_0 \sum_{k=0}^{\infty} (2k-1)!! \frac{(-2\pi i)^{2k}}{(2k)!} \left(\frac{m_2}{m_0}\right)^k u^{2k} \tag{13}$$

(we recall $\mathcal{F}(P_G(f))$ is a Gaussian).

We can see $I_G(f)$ is a ratio of two absolutely convergent power series, so $I_G(f)$ itself can be expressed as an absolutely convergent power series of the form

$$I_G(f)(u) = \sum_{k=0}^{\infty} \frac{(-2\pi i)^k}{k!} a_k u^k$$

where a_k are the moments of the primordial image. Substituting the above three power series into the definition of $I_G(f)$ and considering that

$$(2k-1)!! = \frac{(2k)!}{2^k \cdot k!}$$

we have

$$\sum_{k=0}^{\infty} \frac{(-2\pi i)^k}{k!} m_k u^k = m_0 \sum_{k=0}^{\infty} \frac{(-2\pi^2)^k}{k!} \left(\frac{m_2}{m_0}\right)^k u^{2k} \cdot \sum_{k=0}^{\infty} \frac{(-2\pi i)^k}{k!} a_k u^k.$$

Comparing the terms with the same power of u we obtain, after some algebraic manipulation, the recursive expression for each a_p

$$a_p = \frac{m_p}{m_0} - \sum_{\substack{k=2 \\ k \text{ even}}}^{p} (k-1)!! \cdot \binom{p}{k} \left(\frac{m_2}{m_0}\right)^{k/2} a_{p-k}. \quad (14)$$

Since the primordial image itself (more precisely, its Fourier transform) was proven to be blur invariant, each its moment must be also a blur invariant. If we restrict ourselves to a brightness-preserving blurring, then m_0 itself is an invariant and we obtain from (14) the simplified final form of Gaussian blur invariants

$$B(p) \equiv m_0 a_p = m_p - \sum_{\substack{k=2 \\ k \text{ even}}}^{p} (k-1)!! \cdot \binom{p}{k} \left(\frac{m_2}{m_0}\right)^{k/2} B(p-k), \quad (15)$$

which can be equivalently expressed in a non-recursive form

$$B(p) = \sum_{\substack{k=0 \\ k \text{ even}}}^{p} (k-1)!! \cdot \binom{p}{k} \left(-\frac{m_2}{m_0}\right)^{k/2} m_{p-k}. \quad (16)$$

For the proof of the equivalence of (15) and (16) see Appendix A.

As we already said, $B(p)$ is actually a p-th moment of the primordial image of f. Regardless of f, $B(1) = 0$ because we work with central moments[4] m_p. It always holds $B(2) = 0$ because the second-order moment was used to eliminate the unknown blur parameter σ. Hence, $B(1)$ and $B(2)$ should not be used in the feature vector since they do not carry any information.

Using the image-domain invariants (15) instead of the Fourier domain ones provides higher robustness to noise and is also faster. In practice, we do not need a complete representation of the images in question. Usually a few invariants provide a sufficient discrimination power, so we use the $B(p)$'s up to the certain order Q only. This Q is a user-defined parameter the determination of which should be based on a discrimination analysis of the database images. The choice of Q is always a compromise between the discriminative power and the complexity of the method.

D. Gaussian Blur Invariants in N Dimensions

Let us assume the image domain is a subset of R^N. The centralized N-D Gaussian function has the form

$$G_\Sigma(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp\left(-\frac{1}{2}\mathbf{x}^T \Sigma^{-1} \mathbf{x}\right), \quad (17)$$

where $\mathbf{x} \equiv (x_1, \ldots, x_N)^T$ and Σ is the covariance matrix which determines the shape of the Gaussian. Provided that $N = 2$ and that the covariance matrix of the blur kernel is diagonal, we define the projection operator as

$$P_G(f)(\mathbf{x}) = m_{00} G_S(\mathbf{x}), \quad (18)$$

where

$$S = \text{diag}(m_{20}/m_{00}, m_{02}/m_{00}).$$

The definition of the central geometric moments m_{pq} in two dimensions is analogous to that in one dimension

$$m_{pq} = \int\int (x_1 - c_1)^p (x_2 - c_2)^q f(x_1, x_2) dx_1 dx_2. \quad (19)$$

A visual example of the projection operator is shown in Fig. 2.

(a) (b)

Fig. 2. The original image f (a) and its projection $P_G(f)$ (b).

Similarly to the 1D case (see Theorem 1), the ratio

$$I_G(f)(\mathbf{u}) = \frac{\mathcal{F}(f)(\mathbf{u})}{\mathcal{F}(P_G(f))(\mathbf{u})}$$

is a Gaussian blur invariant. After applying the Taylor expansion, we end up with the following moment invariants analogous to (15)

$$B(p,q) = m_{pq} - \sum_{\substack{k+j=2 \\ k,j \text{ even}}}^{p,q} (k-1)!! \cdot (j-1)!! \cdot \binom{p}{k}\binom{q}{j} \times \left(\frac{m_{20}}{m_{00}}\right)^{k/2} \left(\frac{m_{02}}{m_{00}}\right)^{j/2} B(p-k, q-j) \quad (20)$$

which can be rewritten into a non-recursive form analogous to (16) as

$$B(p,q) = \sum_{\substack{k,j=0 \\ k,j \text{ even}}}^{p,q} (k-1)!! \cdot (j-1)!! \cdot \binom{p}{k}\binom{q}{j} \times \left(-\frac{m_{20}}{m_{00}}\right)^{k/2} \left(-\frac{m_{02}}{m_{00}}\right)^{j/2} m_{p-k,q-j}. \quad (21)$$

For the general case of $N > 2$ see Appendix B.

Note that unlike the Zhang's method, we are not limited to circularly symmetric Gaussian blur kernels but we allow different extent of blur in x_1 and x_2 directions.[5] This may be useful when the horizontal and vertical resolutions of the sensor differ one another. Again, certain invariants are trivial: $B(1,0) = B(0,1) = 0$ due to the centralization, $B(2,0) = B(0,2) = 0$ due to the parameter elimination.[6]

[4]This theory is valid also when using standard non-centralized moments. Then generally $B(1) \neq 0$ but working with central moments is advantageous since it assures the shift invariance.

[5]The Zhang's method could very likely be also generalized to non-isotropic blurs but on the expense of additional time.

[6]If the blur kernel is circularly symmetric, there is only one parameter to be eliminated and we obtain an additional independent invariant $m_{20} - m_{02}$.

E. Translational, Scaling, and Rotational Invariance

Invariance w.r.t. to image translation, scaling and rotation (TSR) is one of the basic requirements we impose on almost any features. If the images are not captured in a fully controlled environment, we always face certain unavoidable unwanted translation/scaling/rotation of the scene.

The Zhang's method is not invariant to translation, scaling, and rotation. This issue was not discussed at all in [65] and the experiments were presented on perfectly registered images only. We believe the Zhang's method could be adapted to be translational invariant but we cannot see any possible extension to scaling and rotation invariance except a time-expensive brute force search.

The invariants $B(p,q)$, introduced in the previous section, are inherently invariant to translation, because they are composed of the central moments of the image. Scaling invariance can be achieved by using the scale-normalized moments instead of the standard moments. This is a commonly used approach in the moment theory (see [63] for details), which is in this case equivalent to dividing each $B(p,q)$ by $m_{00}^{(p+q+2)/2}$.

Since the standard moments change under rotation in a complicated way, the rotation invariance of $B(p,q)$ cannot be achieved readily (let us speak about the 2D case only because handling the rotation in higher dimensions requires special mathematical tools and it is of less practical importance). We will adopt the trick discovered by Flusser [67], who used it for construction of rotation moment invariants (with no relationship to blurring).

First of all, we have to use other moments than the geometric ones, which change under rotation in a simple way. There exist a class of such moments (see [63] for a survey) called *radial moments*. Their common feature is that their 2D basis functions are products of 1D radial polynomials and angular harmonic functions. They use to be complex valued and under the image rotation only their phase is changed (the reader can recognize a clear analogy with the Fourier Shift Theorem). Here we employ so-called *complex moments*

$$c_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x+iy)^p (x-iy)^q f(x,y)dxdy \tag{22}$$

that are linked to the previously used geometric moments as

$$c_{pq} = \sum_{k=0}^{p}\sum_{j=0}^{q}\binom{p}{k}\binom{q}{j}(-1)^{q-j}\cdot i^{p+q-k-j}\cdot m_{k+j,p+q-k-j} \tag{23}$$

and inversely as

$$m_{pq} = \frac{1}{2^{p+q}i^q}\sum_{k=0}^{p}\sum_{j=0}^{q}\binom{p}{k}\binom{q}{j}(-1)^{q-j}\cdot c_{k+j,p+q-k-j}. \tag{24}$$

Note that $c_{pq} = c_{qp}^*$, so only the moments with $p \geq q$ are independent and meaningful to consider. After a transformation into polar coordinates (r,θ), the complex moments obtain the form

$$c_{pq} = \int_0^{\infty}\int_0^{2\pi} r^{p+q+1} e^{i(p-q)\theta} f(r,\theta)d\theta dr. \tag{25}$$

From the last equation we can see that after a coordinate rotation by angle α the complex moment is changed as

$$c'_{pq} = e^{-i(p-q)\alpha}\cdot c_{pq}. \tag{26}$$

The complex moments of the blurred image (1) are in general

$$c_{pq}^{(g)} = \sum_{k=0}^{p}\sum_{j=0}^{q}\binom{p}{k}\binom{q}{j}c_{kj}^{(h)}c_{p-k,q-j}^{(f)}. \tag{27}$$

If the blur kernel h is a circularly symmetric Gaussian (4) then we have for its moments

$$c_{pq}^{(h)} = \begin{cases}(2\sigma^2)^p p! & p = q\\ 0 & p \neq q\end{cases}$$

and Eq. (27) becomes

$$c_{pq}^{(g)} = \sum_{j=0}^{q}\binom{p}{j}\binom{q}{j}j!(2\sigma^2)^j c_{p-j,q-j}^{(f)}, \tag{28}$$

assuming that $p \geq q$.

Now we use the complex moments to derive invariants w.r.t. Gaussian blur in a similar way as the geometric moments were used earlier. Similarly to Eq. (5), we define the projection operator as

$$P_G(f)(x,y) = c_{00}G_s(x,y) \equiv \frac{c_{00}}{2\pi s^2}e^{-\frac{x^2+y^2}{2s^2}}, \tag{29}$$

where

$$s^2 = c_{11}/(2c_{00}).$$

$P_G(f)$ has the same c_{00} and c_{11} as f (and of course $c_{10} = 0$ when working in the centralized coordinates). The other moments of $P_G(f)$ and f are generally different from one another. The following relation shows that the complex moments are "almost" the Taylor coefficients of the Fourier transform of f. Let us make a substitution $U = u + v$ and $V = i(u - v)$. Then

$$\begin{aligned}\mathcal{F}(f)(U,V) &\equiv \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-2\pi i(Ux+Vy)}f(x,y)dxdy\\ &= \sum_{j=0}^{\infty}\sum_{k=0}^{\infty}\frac{(-2\pi i)^{j+k}}{j!k!}c_{jk}u^j v^k.\end{aligned} \tag{30}$$

Using the same substitution, we define the blur invariant in Fourier domain analogously to Theorem 1 as

$$I_G(f)(U,V) = \frac{\mathcal{F}(f)(U,V)}{\mathcal{F}(P_G(f))(U,V)}.$$

Taylor expansion of the denominator is

$$\mathcal{F}(P_G(f))(U,V) = c_{00}\sum_{k=0}^{\infty}\frac{(-4\pi^2)^k}{k!}\left(\frac{c_{11}}{c_{00}}\right)^k u^k v^k.$$

Using the Taylor expansion of all three factors by means of their complex moments and comparing the coefficients of the same powers, we obtain the blur invariants in the image domain

$$K(p,q) = c_{pq} - \sum_{k=1}^{q} k! \binom{p}{k} \binom{q}{k} \left(\frac{c_{11}}{c_{00}}\right)^k K(p-k, q-k)$$
$$= \sum_{k=0}^{q} k! \binom{p}{k} \binom{q}{k} \left(-\frac{c_{11}}{c_{00}}\right)^k c_{p-k,q-k}. \quad (31)$$

Note that $K(q,p) = K(p,q)^*$, $K(1,0) = 0$ when working in the centralized coordinates, and $K(1,1) = 0$ due to the parameter elimination.

The invariants $K(p,q)$ are formally similar to the $B(p,q)$'s (only the moment type was changed). They are actually different but thanks to the link between the geometric and complex moments (23, 24) they generate the same invariant subspace. The principle difference between them (and the reason why we employed the complex moments in this section) is that the $K(p,q)$'s are easy to handle under an image rotation. They change in the same way as the complex moments themselves, i.e.

$$K'(p,q) = e^{-i(p-q)\alpha} \cdot K(p,q). \quad (32)$$

The simplest way to achieve the rotation invariance is to take the magnitudes $|K(p,q)|$ which provide combined invariants but create only an incomplete system. A more sophisticated method is based on the phase cancellation by multiplication of proper invariants, which leads for instance to the invariants[7]

$$K(p,q)K(1,2)^{p-q}.$$

Scale invariance of $K(p,q)$'s can be achieved by the same normalization as in the case of $B(p,q)$'s.

IV. Experiments and a Comparison to the Zhang's Method

The aim of this section is not only to demonstrate the performance of the proposed method but also to compare it to the method by Zhang *et al.* [65]. Comparison to the Zhang's method is highly relevant because both methods have been designed specifically for Gaussian-blurred images, both are theoretically invariant to blur and both should provide good recognition power. There have been proposed no other competitors of these properties in the literature. To make the comparison as fair as possible, we asked the authors of [65] for providing all necessary original codes. Then we implemented our method using the same version of Matlab (R2013a) and always run both on the same computer (Dell Notebook, VOSTRO 1510, Intel, Core2 Duo CPU, 4GB RAM, Windows 8, 32-bit) and on the same test images. Since the Zhang's method can compare only images of the same size, we kept this condition in all experiments.

In some experiments we included also two other method into the comparison – image cross-correlation and Local phase quantization (LPQ) [52]–[55]. The cross-correlation is of course not blur invariant, so the comparison with it shows what is the actual benefit of the blur invariance property. LPQ is a representative of methods acting in the Fourier domain. LPQ is invariant to general centrosymmetric blur, it does not employ the parametric form of the PSF at all. The main idea is that a centrosymmetric PSF does not change the phase of the Fourier transform in certain neighborhood of the origin. The Fourier transform is windowed and its phase in a close neighborhood of the origin is quantized and taken as a local descriptor. We originally used the LPQ code provided by the authors which we later improved to reach better performance. Some other method had been compared to the Zhang's distance (ZD) already in [65]. The reader can find there a comparison to standard Euclidean distance, the Gopalan method [62], and centrosymmetric blur invariants [8]. Since the ZD had been evaluated as the best performing method among these, we did not incorporate these comparative methods into our current tests.

The first set of the experiments only illustrates the properties of both methods, which already were proved theoretically. The core experiments can be found in the second set, where statistically significant comparison of the success rate and the time complexity is presented.

TABLE I
The Values of ZD and ID in the Case of Simulated Gaussian Blur

σ	Filter size	ZD	ID
0	1	0	0
1	9	0.112	2e-19
2	17	0.005	1e-19
3	25	0.009	3e-19
4	33	0.012	4e-19
5	41	0.014	6e-19
6	49	0.016	2e-19
7	57	0.017	2e-19

A. Blur Invariance Property

As we expected, both methods actually exhibit high invariance w.r.t. a "perfect" (i.e. computer-generated) Gaussian blur (see Table I). We changed the blur parameter σ from 0 to 7 and calculated both the Zhang's distance ZD and the Euclidean distance in the space of the invariants (31) between the blurred image and the original. We refer to the distance in the space of the invariants as the *invariant distance* ID. Both distances in this experiment are reasonably small although not zero. The non-zero values appear because the sampled Gaussian does not fulfil exactly the assumption. Since larger sampled Gaussians are more accurate, we observe that the error sometimes decreases as the blur size increases, although one might expect an opposite relation. For comparison, we also calculated the distances between several *different* originals, which is by two orders higher. The test images were of the size 160×160 pixels (see Fig. 3 for an example).

B. Shift, Rotation, and Scaling Invariance

Here we experimentally verified the theoretical knowledge that our method provides the invariance w.r.t. these three elementary geometric transformations while the Zhang's method

[7]This set can be proven to be complete and independent provided that $K(1,2) \neq 0$. Other choices are also possible and lead to equivalent invariants.

Fig. 3. One of the original images (160×160 pixels) used in the tests.

TABLE II
THE DISTANCE BETWEEN THE ORIGINAL AND ITS SHIFTED COPY

Shift (in pixels)	ZD	ID
0	0	0
5	435	2e-19
10	548	2e-19
15	607	0
20	630	3e-19
25	639	2e-20
30	644	2e-20
35	647	7e-20

TABLE III
THE DISTANCE BETWEEN THE ORIGINAL AND ITS ROTATED COPY

Rotation angle [deg]	ZD	ID
0	0	0
10	389	7e-10
20	452	2e-9
30	462	9e-9
40	443	1e-8
50	416	1e-8
60	380	5e-9
70	339	2e-9
80	291	1e-10
90	230	1e-19

TABLE IV
THE DISTANCE BETWEEN THE ORIGINAL AND ITS SCALED COPY

Scaling factor	ZD	ID
1	0	0
0.8	589	1e-6
0.6	716	1e-6
0.4	774	3e-6
0.2	829	1e-5

is sensitive to the particular image position, orientation and size. It is worth mentioning how sensitive the ZD is to the shift. As the shift approaches 5 pixels, the Zhang distance between the shifted images is comparable to the distance between two completely different images (see Table II). The same is true for the scaling and rotation, too. Even a small rotation/scaling harms the ZD substantially (see Table III and Table IV). The sensitivity to a shift is also a weakness of the LPQ method. The Fourier phase is changed when the image has been shifted, so the LPQ feature is changed as well. There exist rotation and scale invariant modifications of LPQ but no shift invariant version has been reported.

TABLE V
THE DISTANCE BETWEEN THE ORIGINAL AND ITS CONTRAST-CHANGED COPY

Contrast factor	ZD	ID
1	0	0
0.9	90	5e-10
0.8	180	2e-10
0.7	270	2e-10
0.6	359	4e-10
0.5	449	0
0.4	538	2e-10
0.3	625	4e-9
0.2	708	2e-10
0.1	783	2e-10

C. Invariance to Contrast Stretching

This easy test verified that the invariants, when normalized by m_{00}, are invariant also to a contrast stretching of the form $g(x, y) = af(x, y)$, $a > 0$. The Zhang's method interprets low contrast as a blur due to lower values of the Laplacian and blurs the more contrast image before the distance is calculated. This leads to an inaccuracy of computation of the ZD, which of course depends on the parameter a (see Table V for illustration). However, this problem of the ZD can easily by resolved by normalizing the images to the same graylevel variance (which, on the other hand, would increase the time complexity).

D. Robustness to Noise

Robustness to additive noise is an important requirement imposed on any features since in reality the noise is unavoidable. When taking a picture in low light, we use high ISO and/or long exposure. Both amplifies the background noise, which is present in any electronic system, such that the noise energy may be even higher than that of the signal. Particularly compact cameras and cell-phone cameras with small-size chips suffer from this kind of noise, along with an omnipresent thermal noise. Although the camera noise contains also a Poisson component, it is commonly modelled as a white Gaussian noise.

First, we added the noise of SNR from 50 dB to -5 dB into the image (see Fig. 4 for some examples), and calculated both ID and ZD from the original. On each noise level, we run the experiment 10 times and the mean values are presented in Table VI. The invariant method is more robust because the moments are defined as integrals, which basically "averages" the noise and decreases its impact on the feature values. On the other hand, the Zhang distance is very sensitive. This is due to its first stage when the image blur level is estimated by measuring the energy in the high-pass band. The noise dominates the image on high frequencies and contributes a lot to this measure. Hence, the blurred image with heavy noise may often be considered "sharper" than the clear image and the method blurs it again to bring it (seemingly) to the same blur level.

We measured the robustness also on real noise. We took a series of photographs in low-light conditions to introduce an observable camera noise. Each of four scenes used here was

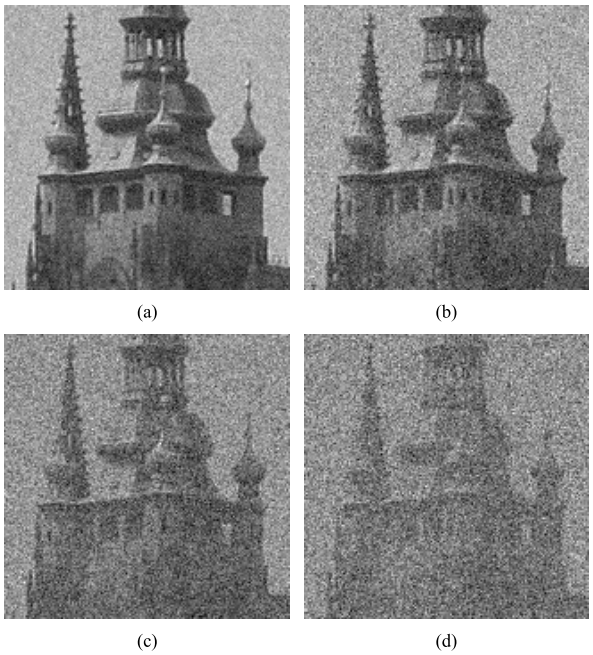


Fig. 4. Examples of the images with simulated noise used in the test. SNR = 10 dB (a), SNR = 5 dB (b), SNR = 0 dB (c), and SNR = −5 dB (d).

TABLE VI

THE DISTANCE BETWEEN THE ORIGINAL AND ITS NOISY VERSION – SIMULATED NOISE

SNR [dB]	ZD	ID
50	1.1	1e-8
40	3.6	3e-8
30	11.0	1e-7
20	31.3	3e-7
10	71.0	1e-6
5	98.7	1e-6
0	135.9	9e-6
-5	193.0	3e-5

TABLE VII

THE DISTANCE BETWEEN THE ORIGINAL AND ITS NOISY VERSION – REAL NOISE

Scene	ZD	ID
1	2.39	2e-6
2	2.07	2e-6
3	1.76	7e-7
4	1.35	4e-6

taken by a multi-shot sequence of 20 frames. The estimated SNR in each frame is about 30 dB. The "clear" image was obtained by a time-averaging of the noisy frames, since it was not possible to take it directly. Such an image is not actually noise-free but the noise is suppressed significantly. For each scene, we calculated both ID and ZD between the "clear" image and each noisy frame. The mean values for each scene are presented in Table VII. Considering that the ideal distance value should be always zero, these results are consistent with those obtained on simulated noise and confirm the better robustness of the ID.

Fig. 5. Sample "clear" images of the challenging database. The database consists of very similar faces. Downloaded from the CASIA HFB dataset.

E. Image Recognition Against Public Databases

The main purpose of ZD and ID is to use them in recognition of Gaussian-blurred images w.r.t. a given database of clear images. As soon as the query image is provided, both ZD and ID look for exactly the same image (up to the blurring and the contrast change) in the database. This recognition should be reliable and fast enough. These methods do not tolerate other differences such as nonlinear deformations, object pose, facial expression, etc. They are inappropriate in the cases where such situation may occur. Since the "image classes" are defined by single representatives, the classification by minimum distance is applied most often.[8]

First of all, we used LIVE and CSIQ databases [68], [69], which were used already in [65]. To our best knowledge, these two databases are the only public datasets containing Gaussian-blurred images. The CSIQ database contains 30 clear images of common urban, animal and landscape scenes and five blurred instances of various extent of each of them. The LIVE database contains similar data but only some of the images are available along with their blurred versions. To reach higher statistical significance, we mixed both databases together. We resampled all images to 128×128 pixels, used 59 clear images as training samples and classify all 324 blurred images by ID, ZD and LPQ. The success rate of all three methods was 100%. This is because the training images

[8]This is, however, not a restriction imposed by ZD/ID themselves. If the training set contained more samples, we could apply k-NN or SVM classifiers.

Fig. 6. Sample test images degraded by heavy blur and noise ($\sigma = 5$ and SNR $= 0$ dB).

are visually clearly different and therefore the blur introduced into the query image does not cause serious problems. Concerning the time, ID was the fastest, LPQ was about ten times slower and ZD was more than 1000 times slower than ID.

For the second experiment we deliberately used a database which is composed of similar images, which are even difficult to distinguish visually. In such a case even a mild blur and noise might result in a number of misclassifications. Although the tested methods have not been developed as specialized face recognition methods, we used facial database for two reasons– it contains very similar images and blur is often present in face recognition applications. We selected 100 face images of 100 persons from the CASIA heterogeneous face biometrics (HFB) database [70], [71]. They are all frontal views with a neutral expression, without a head pose, and with the face being cropped (see Fig. 5). We successively took each image of the dataset, blurred it by a Gaussian blur, added a noise, and classified the image by the minimum distance rule against the database of the "clear" images (see Fig. 6 for the samples of the most degraded images). We did this for various amount of blur and noise and measured the success rate. For each blur and noise level we generated ten instances of the query image. Hence, we classified 36,000 images altogether.

The results of all three methods are summarized in a form of the "blur size – SNR" matrices in Table VIII. While for low amount of blur and noise all methods work very well, the performance of ZD drops as the image degradations increase (check the lower right part of the matrix). The performance

TABLE VIII

RECOGNITION RATE (IN %) OF BLURRED AND NOISY FACES BY ZD, LPQ AND ID

SNR [dB]/σ	0	1	2	3	4	5
Method			ZD			
60	100	100	100	100	100	100
40	100	100	100	100	100	100
20	100	100	100	100	97	74
10	100	100	100	82	48	30
5	100	100	100	74	41	26
0	100	100	98	60	35	21
Method			LPQ			
60	100	100	100	100	100	96
40	100	100	100	100	100	95.9
20	100	100	100	100	100	95.6
10	100	100	100	100	99.9	93.6
5	100	100	100	100	99.6	91.6
0	100	100	99.9	99.5	93	69.9
Method			ID			
60	100	100	100	100	100	100
40	100	100	100	100	100	100
20	100	100	100	100	100	100
10	100	100	100	100	100	100
5	100	100	100	100	99.9	99.8
0	100	96	95	95	94	94

of the LPQ is comparable to that of the ID except the last column corresponding to the largest blur ($\sigma = 5$), where the ID performs much better.

The success rate of the ID is almost 100% in all cases except SNR $= 0$ dB, which is mainly due to the guaranteed invariance of the ID w.r.t. blur and good robustness to additive noise.

We also measured the time needed for recognition of one image (this time does not depend on the particular configuration of the blur and noise). The Zhang's method requires 1500 seconds, the LPQ 0.22 second and the proposed method works in 0.05 second only. This difference in complexity is mainly caused by the fact that the invariant values as well as the LPQ descriptors of the database images are calculated only once and used repeatedly, while the Zhang's distance is calculated "from scratch" for each pair. The LPQ feature is of a high dimension comparing to the invariants. When calculating ID, only the invariants up to the order 8 were used, while the LPQ feature in the basic version has the same size as the image itself. Since the features are supposed to be stored in the database for a repeated usage, this high dimensionality makes the LPQ method inefficient in terms of the memory usage. The LPQ features can be quantized and compressed into a histogram only which speeds up the recognition and improves the memory usage (we actually used this trick in our experiment), but the dimensionality is still at least by one order higher than the dimensionality of the blur invariants. On the other hand, thanks to its redundancy, the LPQ achieves relatively good recognition rates.

F. Matching of Blurred Templates - Simulated Blur

In this experiment we tested the performance in the template matching, which is a particular classification problem we often face in practice. Assuming that we have a large clear image of a scene and a blurred template, the task is to localize this template in the clear image. We again tested both ID and ZD.

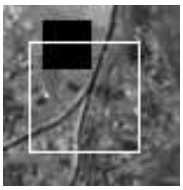


Fig. 7. Explanation of the boundary effect. The inside pixels near the template boundary (white square) are affected by the pixels from the outside of the template if the scene is blurred. The extent of this effect depends on the blur size (black square).

For a comparison we included also the cross-correlation (CC) as a "gold standard" method which has been traditionally used in matching of non-blurred templates. Since the testing of each possible template location is very time consuming, we used all three methods in a hierarchical coarse-to-fine implementation. On the coarse level, we shifted the template by the step of 4 pixels in each direction. On the fine level, we searched a 9×9 neighborhood of the "best" location found on the coarse level. Provided that the horizontal and vertical localization errors are independent and both have the same normal distribution, the absolute localization error has a Rayleigh distribution. We estimated the mean values and standard deviations of the localization error of all three methods, which illustrates the accuracy. Since these parameters might be influenced by few big errors, we also calculated the number of "correct hits", which may serve as another (and probably more relevant) accuracy measure. We marked the position of the template found by the algorithm as a hit, if its localization error was less or equal to one pixel in each direction.

Note that in template matching, when the blurred templates have been extracted from a large scene, we always face a *boundary effect*. This means there is a strip along the template boundary where the convolution model is not valid (even if the blur has been introduced artificially) because the pixels laying outside the template also contribute to the intensity values inside this strip due to the blurring kernel (see Fig. 7). The boundary effect is the main source of errors in a noise-free case.

We took a clear image of the size 256×256, blurred it by a 13×13 Gaussian of $\sigma = 2$ and randomly selected 30 templates of the size 32×32. These templates were searched in the clear image. We used the invariants up to the order six. The results of the matching in terms of the accuracy and computational time are summarized in Table IX. We can see that the accuracy of both ID and ZD are excellent, so both methods are stable w.r.t. the boundary effect. The ZD yields even better localization error than ID because it uses a complete information about the template while the invariants work with highly compressed information. On the other hand, ID is more than 20 times faster than ZD. The CC was much faster than ID but its accuracy was very low because of the blurring. The time measurement for one template includes a complete "scan" of the scene including invariant and distance calculation for each tested position and search for the minimum distance. Overheads (reading of the images, generating blur kernel, blurring the image, template selection, etc.) are common for all methods and were not included into the measurement.

TABLE IX
MATCHING OF BLURRED NOISE-FREE TEMPLATES

Method	Mean error	Std	Mean time [s]	Correct hits
CC	42.53	22.22	1.29	23
ZD	0.16	0.08	831.5	30
ID	0.39	0.20	34.6	30

TABLE X
MATCHING OF BLURRED AND NOISY TEMPLATES

Method	Mean error	Std	Mean time [s]	Correct hits
CC	41.24	21.55	1.31	20
ZD	43.99	22.98	825.1	15
ID	0.90	0.47	33.1	28

(a) (b) (c)

Fig. 8. The test image "Fox": (a) original, (b) blurred image, (c) blurred and noisy image, SNR = 10 dB.

Then we repeated the same experiment with the same setting and with the same templates but we added a Gaussian white noise of SNR = 10 dB into the blurred image (see Fig. 8). As can be seen from Table X, the results changed dramatically. The ID still provides 28 correct hits and the mean error less than one, while the ZD was even worse than the CC. The explanation of the difference in robustness is the same as that given in Section IV.D. The time complexity is basically the same as in the first experiment.

We also studied the behavior of the invariants under variable blur and template size and on various noise levels. In all following experiments we used the invariants up to the order 6. First, we fixed the template size to 32×32 while the Gaussian σ increased from 1 to 5 by a sampling step 0.5. In each parameter setting we matched 30 randomly chosen templates. This experiment was run five times and the means of the correct hits are shown in a graph in Fig. 9. Then we run the whole experiment again with the same templates corrupted by a noise of SNR = 0 dB. As one may expect, the results are much worse namely in case of small blur (see Fig. 9). In case of heavy blur, the main source of errors is a boundary effect and the influence of noise is not so significant.

In a complementary experiment, we fixed $\sigma = 2$ and changed the SNR only. The means of the correct hits over 30 runs are shown in Fig. 10. All templates were matched correctly for SNR > 25 dB. As the SNR decreases, the number of errors increases, reaching 53% if SNR = 0 dB.

In the last experiment, we investigated the influence of the template size on the success rate and the computation time of the ID. We fixed $\sigma = 2$ while the template size changed from 64×64 to 8×8 pixels. To make the comparison fair,

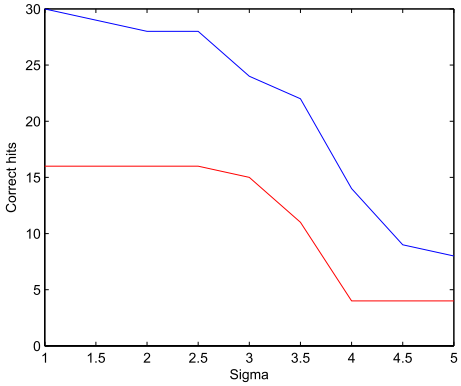


Fig. 9. The number of correctly located templates as a function of the blur size in the noise-free case (blue curve) and in the noisy case (SNR = 0 dB, red curve).

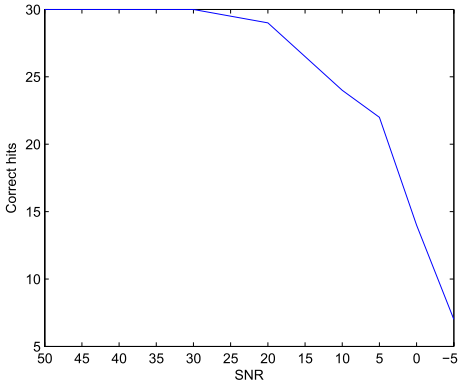


Fig. 10. The number of correctly located blurred templates ($\sigma = 2$) as a function of SNR.

only the positions of the 64×64 templates were selected randomly. The smaller templates were obtained by cropping of the largest ones. As one can expect, both the success rate and the computation time decrease along with the decreasing template size (see Table XI, the numbers are means over 5 runs of experiment with 30 templates). The main source of the errors is the boundary effect (which is more significant in small templates since the blur size has been fixed). The time complexity is given by the complexity of moment computation; the calculation of the invariants and the matching itself do not depend on the template size. However, the decrease of the computation time is mild comparing to the rapid increase of the success rate. Taking these two criteria into account simultaneously, one may conclude that in practice large templates should be preferred since they provide better success/speed gain than the small ones.

G. Matching of Blurred Templates - Real Blur

Finally, we performed a template matching experiment on astronomical images degraded by real atmospheric turbulence blur. We employed four images of the spot in the solar photosphere taken by a telescope with a CCD camera in a visible spectral band (the venue: Observatory Ondrejov, Czech Republic; wavelength: $\lambda \doteq 590$ nm). Since the time interval between the two consecutive acquisitions was only few seconds, the scene can be considered still and the images are almost perfectly registered. As the atmospheric conditions changed between the acquisitions, the amount of blur in individual images vary from one another. We sorted the images according to their blur level by means of the algorithm which compares the energy in low-pass and high-pass wavelet transform bands [66]. The ordered sequence can be seen (and visually checked) in Fig. 11. The size of each image is 256×256 pixels. The first image is relatively sharp while the other three images, particularly the last one, are noticeably blurred. The blur kernel is believed to be approximately Gaussian (an experimental validation of this assumption can be found for instance in [72]). Mild additive noise is also present in all images, its estimated SNR is about 30 dB.

By the the four methods used in the previous experiments (CC, ZD, LPQ, and ID), we matched 30 randomly chosen 32×32 templates extracted from the first "clear" image against each of the other three images. The maximum order of the invariants used was six. The coarse-to-fine matching algorithm was used with the coarse step 8 pixels and with a 16×16 search area on the fine level. For each template, we consider any possible position, we did not apply any restricted search area. This is equivalent to the classification of 30 query images against a database of $3(256 - 32)^2 = 150528$ images.

As one can see from Table XII, the results are consistent with those we achieved on simulated blurring. The CC localization accuracy is the worst one because of the blur. The Zhang's distance provides slightly worse accuracy than the invariants. The reason is the presence of noise. Even if the noise is very mild, ZD is highly sensitive to it for the reasons

TABLE XI
MATCHING OF BLURRED AND NOISY TEMPLATES BY ID

Template size	Mean time [s]	Correct hits
64×64	30.1	30
32×32	24.8	27
16×16	23.8	16
8×8	23.2	2

TABLE XII
TEMPLATE MATCHING IN ASTRONOMICAL IMAGES

Image	Mean error	Standard deviation	Correct hits
	Cross-correlation (CC)		
(b)	7.30	3.82	29
(c)	7.29	3.81	29
(d)	7.28	3.80	28
	Zhang distance (ZD)		
(b)	3.73	1.95	28
(c)	3.71	1.94	28
(d)	3.50	1.82	28
	Local phase quantization (LPQ)		
(b)	0.87	0.46	30
(c)	0.84	0.44	30
(d)	0.90	0.47	30
	Invariant distance (ID)		
(b)	0.88	0.45	30
(c)	0.90	0.47	30
(d)	0.85	0.44	30

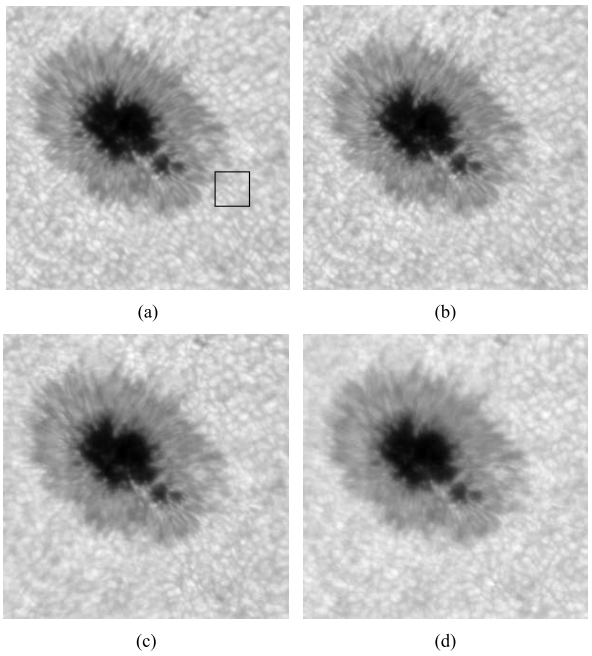


Fig. 11. Four images of the sunspot blurred by atmospheric turbulence blur of various extent. The images are ordered from the less to the most blurred one. One template is depicted in the first image to illustrate its size.

which we already explained in Section IV.D. Both ID and LPQ provides a perfect localization accuracy in this experiment. This is admirable when considering that each template was tested against 150,528 possible positions and that many of them have a very similar visual appearance.

V. Conclusion

We proposed new invariants w.r.t. Gaussian blur, both in frequency and image domains. We showed the performance of the new method in object recognition and in matching of blurred and noisy templates. Comparing to the Zhang's method [65], which has been the only Gaussian-blur invariant metric so far, the proposed method is significantly faster and more robust to additive noise while its recognition rate in noise-free cases is fully comparable to the Zhang's distance. An additional benefit of the new method is that it can be easily made invariant to translation, rotation, scale, and contrast of the image, which is very important in many applications and which is not the case of the Zhang's method. Last but not least, our method handles also an anisotropic Gaussian blur and is even able to compare images of different sizes.

Appendix A

The proof of the equivalence of Eqs. (15) and (16) is due to induction on p. For $p = 0, 1, 2$ the equivalence holds well. Now we show the induction step. To avoid the necessity of discrimination between even and odd p's, we use a re-indexing in the sums. Introducing $K = [p/2]$ and, for simplicity, $m = m_2/m_0$ we have for Eq. (15)

$$\begin{aligned}
B(p) &= m_p - \sum_{k=1}^{K}(2k-1)!! \cdot \binom{p}{2k} m^k B(p-2k) \\
&= m_p - \sum_{k=1}^{K}(2k-1)!! \cdot \binom{p}{2k} m^k \sum_{j=0}^{K-k}(2j-1)!! \\
&\quad \times \binom{p-2k}{2j}(-m)^j m_{p-2k-2j} \\
&= m_p - \sum_{k=1}^{K}\sum_{j=0}^{K-k}(-1)^j \frac{p!}{2^{k+j}k!j!(p-2k-2j)!} m^{k+j} \\
&\quad \times m_{p-2k-2j} \\
&= m_p - \sum_{k=1}^{K}\sum_{j=k}^{K}(-1)^{j-k} \frac{p!}{2^j k!(j-k)!(p-2j)!} m^j \\
&\quad \times m_{p-2j} \\
&= m_p - \sum_{j=1}^{K}\sum_{k=1}^{j}(-1)^{j-k} \frac{p!}{2^j k!(j-k)!(p-2j)!} m^j \\
&\quad \times m_{p-2j} \\
&= m_p - \sum_{j=1}^{K}(-1)^j \frac{p!}{2^j (p-2j)!} m^j m_{p-2j} \sum_{k=1}^{j} \frac{(-1)^k}{k!(j-k)!}.
\end{aligned}$$

Since

$$\sum_{k=1}^{j}(-1)^k \cdot \binom{j}{k} = -1$$

for any j, we obtain

$$\begin{aligned}
B(p) &= m_p + \sum_{j=1}^{K}(2j-1)!!\binom{p}{2j}(-m)^j m_{p-2j} \\
&= \sum_{j=0}^{K}(2j-1)!!\binom{p}{2j}(-m)^j m_{p-2j},
\end{aligned}$$

which exactly matches Eq. (16).

Appendix B

Let us introduce a vector notation

$$\begin{aligned}
|\mathbf{p}| &\equiv \sum_{i=1}^{N} p_i, \quad \binom{\mathbf{p}}{\mathbf{k}} \equiv \prod_{i=1}^{N}\binom{p_i}{k_i}, \\
\mathbf{p}^{\mathbf{k}} &\equiv \prod_{i=1}^{N} p_i^{k_i}, \quad \mathbf{p}!! \equiv \prod_{i=1}^{N} p_i!!, \\
\mathbf{0} &\equiv (0, 0, \ldots, 0), \quad \mathbf{1} \equiv (1, 1, \ldots, 1).
\end{aligned}$$

The moment of function $f(\mathbf{x})$ is given as

$$m_{\mathbf{p}} = \int (\mathbf{x} - \mathbf{c})^{\mathbf{p}} f(\mathbf{x}) d\mathbf{x}. \tag{33}$$

The moment of a Gaussian kernel with a diagonal covariance matric $\Sigma = \mathrm{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_N^2)$ is, in the case that all elements of $\mathbf{p}$ are even, given as

$$m_{\mathbf{p}} = \sigma^{\mathbf{p}}(\mathbf{p} - \mathbf{1})!! \tag{34}$$

where $\sigma \equiv (\sigma_1, \sigma_2, \ldots, \sigma_N)$. All other moments are zero. Then we can define an N-dimensional projection operator

$$P_G(f)(\mathbf{x}) = m_{\mathbf{0}} G_S(\mathbf{x}),$$

where

$$S = \mathrm{diag}(\mathbf{m}_2 / m_{\mathbf{0}})$$

and

$$\mathbf{m}_2 \equiv (m_{20\ldots0}, m_{02\ldots0}, \ldots, m_{00\ldots2}).$$

The N-D versions of the invariants (15) and (16) are

$$B(\mathbf{p}) = m_{\mathbf{p}} - \sum_{\substack{\mathbf{k}=\mathbf{0} \\ 0<|\mathbf{k}|}}^{\mathbf{p}} (\mathbf{k}-\mathbf{1})!! \cdot \binom{\mathbf{p}}{\mathbf{k}} (\mathbf{m}_2/m_{\mathbf{0}})^{\mathbf{k}} B(\mathbf{p}-\mathbf{k})$$

$$= \sum_{\mathbf{k}=\mathbf{0}}^{\mathbf{p}} (\mathbf{k}-\mathbf{1})!! \cdot \binom{\mathbf{p}}{\mathbf{k}} (-1)^{|\mathbf{k}|} (\mathbf{m}_2/m_{\mathbf{0}})^{\mathbf{k}} m_{\mathbf{p}-2\mathbf{k}}, \quad (35)$$

where the summation goes over those multi-indices $\mathbf{k}$ *all elements of which are even*.

We can do the same even if Σ is not diagonal but the directions of its eigenvectors must be known. The formula for the invariants would, however, look much more complicated. If the eigenvectors of Σ are not known, we cannot properly "rotate" the image, the projection operators cannot be defined and the derivation of the invariants fails.

ACKNOWLEDGMENT

We thank the authors of the paper [65] for providing the implementation of their method for comparison, Dr. Stanislava Šimberova for providing the astronomical test images used in the template matching experiment, and Dr. Jaroslav Kautsky for his advice concerning numerical implementation of moments. Sajad Farokhi thanks the Czech Academy of Sciences for the post-doctoral scholarship covering his one-year stay at the Institute of Information Theory and Automation in Prague.

REFERENCES

[1] B. Honarvar, R. Paramesran, and C.-L. Lim, "Image reconstruction from a complete set of geometric and complex moments," *Signal Process.*, vol. 9 8, pp. 224–232, May 2014.

[2] B. Honarvar and J. Flusser, "Image deconvolution in moment domain," in *Moments and Moment Invariants—Theory and Applications*, G. A. Papakostas, Ed. Xanthi, Greece: Science Gate Publishing, 2014, pp. 111–125.

[3] A. S. Carasso, "The APEX method in image sharpening and the use of low exponent Lévy stable laws," *SIAM J. Appl. Math.*, vol. 63, no. 2, pp. 593–618, 2003.

[4] J. H. Elder and S. W. Zucker, "Local scale control for edge detection and blur estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 7, pp. 699–716, Jul. 1998.

[5] W. Zhang and W.-K. Cham, "Single-image refocusing and defocusing," *IEEE Trans. Image Process.*, vol. 21, no. 2, pp. 873–882, Feb. 2012.

[6] F. Xue and T. Blu, "A novel SURE-based criterion for parametric PSF estimation," *IEEE Trans. Image Process.*, vol. 24, no. 2, pp. 595–607, Feb. 2015.

[7] J. Flusser, T. Suk, and S. Saic, "Recognition of blurred images by the method of moments," *IEEE Trans. Image Process.*, vol. 5, no. 3, pp. 533–538, Mar. 1996.

[8] J. Flusser and T. Suk, "Degraded image analysis: An invariant approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 6, pp. 590–603, Jun. 1998.

[9] V. Ojansivu, "Blur invariant pattern recognition and registration in the Fourier domain," Ph.D. dissertation, Dept. Electr. Inf. Eng., Univ. Oulu, Oulu, Finland, 2009.

[10] Y. Bentoutou, N. Taleb, K. Kpalma, and J. Ronsin, "An automatic image registration for applications in remote sensing," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 9, pp. 2127–2137, Sep. 2005.

[11] Z. Liu, J. An, and L. Li, "A two-stage registration algorithm for oil spill aerial image by invariants-based similarity and improved ICP," *Int. J. Remote Sens.*, vol. 32, no. 13, pp. 3649–3664, 2011.

[12] X. Q. Huang, Y.-M. Xiong, M. Z. W. Liao, and W. F. Chen, "Accurate point matching based on combined moment invariants and their new statistical metric," in *Proc. Int. Conf. Wavelet Anal. Pattern Recognit. (ICWAPR)*, Nov. 2007, pp. 376–381.

[13] Y. Bentoutou, N. Taleb, M. C. El Mezouar, M. Taleb, and J. Jetto, "An invariant approach for image registration in digital subtraction angiography," *Pattern Recognit.*, vol. 35, no. 12, pp. 2853–2865, Dec. 2002.

[14] Y. Bentoutou and N. Taleb, "Automatic extraction of control points for digital subtraction angiography image enhancement," *IEEE Trans. Nucl. Sci.*, vol. 52, no. 1, pp. 238–246, Feb. 2005.

[15] Y. Bentoutou and N. Taleb, "A 3-D space–time motion detection for an invariant image registration approach in digital subtraction angiography," *Comput. Vis. Image Understand.*, vol. 97, no. 1, pp. 30–50, Jan. 2005.

[16] Y. Zhang, C. Wen, and Y. Zhang, "Estimation of motion parameters from blurred images," *Pattern Recognit. Lett.*, vol. 21, no. 5, pp. 425–433, May 2000.

[17] Y. Zhang, C. Wen, Y. Zhang, and Y. C. Soh, "Determination of blur and affine combined invariants by normalization," *Pattern Recognit.*, vol. 35, no. 1, pp. 211–221, Jan. 2002.

[18] J. Lu and Y. Yoshida, "Blurred image recognition based on phase invariants," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E82-A, no. 8, pp. 1450–1455, 1999.

[19] X.-J. Shen and J.-M. Pan, "Monocular visual servoing based on image moments," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E87-A, no. 7, pp. 1798–1803, 2004.

[20] B. Mahdian and S. Saic, "Detection of copy–move forgery using a method based on blur moment invariants," *Forensic Sci. Int.*, vol. 171, nos. 2–3, pp. 180–189, Sep. 2007.

[21] B. Mahdian and S. Saic, "Detection of near-duplicated image regions," in *Computer Recognition Systems 2* (Advances in Soft Computing), vol. 45. Berlin, Germany: Springer, 2007, pp. 187–195.

[22] L. Li and G. Ma, "Recognition of degraded traffic sign symbols using PNN and combined blur and affine invariants," in *Proc. 4th Int. Conf. Natural Comput. (ICNC)*, vol. 3. Oct. 2008, pp. 515–520.

[23] L. Li and G. Ma, "Optimizing the performance of probabilistic neural networks using PSO in the task of traffic sign recognition," in *Proc. 4th Int. Conf. Intell. Comput. Adv. Intell. Comput. Theories Appl. Aspects Artif. Intell. (ICIC)*, vol. LNAI 5227. 2008, pp. 90–98.

[24] Y. Zhang, C. Wen, and Y. Zhang, "Neural network based classification using blur degradation and affine deformation invariant features," in *Proc. 13th Int. Florida Artif. Intell. Res. Soc. Conf. (FLAIRS)*, 2000, pp. 76–80.

[25] C. Guang-Sheng and Z. Peng, "Dynamic wood slice recognition using image blur information," *Sens. Actuators A, Phys.*, vol. 176, pp. 27–33, Apr. 2012.

[26] J. Flusser, T. Suk, and B. Zitová, "On the recognition of wood slices by means of blur invariants," *Sens. Actuators A, Phys.*, vol. 198, pp. 113–118, Aug. 2013.

[27] P. Zhao, "Dynamic timber cell recognition using two-dimensional image measurement machine," *AIP Rev. Sci. Instrum.*, vol. 82, no. 8, p. 083703, 2011.

[28] J. Flusser and B. Zitová, "Combined invariants to linear filtering and rotation," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 13, no. 8, pp. 1123–1136, 1999.

[29] J. Flusser, B. Zitová, and T. Suk, "Invariant-based registration of rotated and blurred images," in *Proc. Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jun. 1999, pp. 1262–1264.

[30] B. Zitová and J. Flusser, "Estimation of camera planar motion from blurred images," in *Proc. Int. Conf. Image Process. (ICIP)*, vol. 2. 2002, pp. II-329–II-332.

[31] T. Suk and J. Flusser, "Combined blur and affine moment invariants and their use in pattern recognition," *Pattern Recognit.*, vol. 36, no. 12, pp. 2895–2907, Dec. 2003.

[32] Y. Li, H. Chen, J. Zhang, and P. Qu, "Combining blur and affine moment invariants in object recognition," *Proc. SPIE* vol. 5253, p. 165, Sep. 2003.

[33] R. Palaniappan, M. P. Paulraj, S. Yaacob, and M. S. Z. Azalan, "A simple sign language recognition system using affine moment blur invariant features," in *Proc. Int. Postgraduate Conf. Eng. (IPCE)*, 2010, p. 5.
[34] Y. Gao, H. Song, X. Tian, and Y. Chen, "Identification algorithm of winged insects based on hybrid moment invariants," in *Proc. 1st Int. Conf. Bioinformat. Biomed. Eng. (iCBBE)*, vol. 2. Jul. 2007, pp. 531–534.
[35] H. Ji, J. Zhu, and H. Zhu, "Combined blur and RST invariant digital image watermarking using complex moment invariants," in *Proc. 2nd Int. Conf. Signals, Circuits Syst. (SCS)*, Shanghai, China, Nov. 2008, pp. 1–6.
[36] H. Zhang, H. Shu, G. N. Han, G. Coatrieux, L. Luo, and J. L. Coatrieux, "Blurred image recognition by Legendre moment invariants," *IEEE Trans. Image Process.*, vol. 19, no. 3, pp. 596–611, Mar. 2010.
[37] C.-Y. Wee and R. Paramesran, "Derivation of blur-invariant features using orthogonal Legendre moments," *IET Comput. Vis.*, vol. 1, no. 2, pp. 66–77, Jun. 2007.
[38] X. Dai, H. Zhang, H. Shu, and L. Luo, "Image recognition by combined invariants of Legendre moment," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Jun. 2010, pp. 1793–1798.
[39] X. Dai, H. Zhang, H. Shu, L. Luo, and T. Liu, "Blurred image registration by combined invariant of Legendre moment and Harris–Laplace detector," in *Proc. 4th Pacific-Rim Symp. Image Video Technol. (PSIVT)*, Nov. 2010, pp. 300–305.
[40] H. Zhu, M. Liu, H. Ji, and Y. Li, "Combined invariants to blur and rotation using Zernike moment descriptors," *Pattern Anal. Appl.*, vol. 13, no. 3, pp. 309–319, Aug. 2010.
[41] B. Chen, H. Shu, H. Zhang, G. Coatrieux, L. Luo, and J. L. Coatrieux, "Combined invariants to similarity transformation and to blur using orthogonal Zernike moments," *IEEE Trans. Image Process.*, vol. 20, no. 2, pp. 345–360, Feb. 2011.
[42] H. Ji and H. Zhu, "Degraded image analysis using Zernike moment invariants," in *Proc. Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2009, pp. 1941–1944.
[43] Q. Li, H. Zhu, and Q. Liu, "Image recognition by combined affine and blur Tchebichef moment invariants," in *Proc. 4th Int. Conf. Image Signal Process. (CISP)*, Oct. 2011, pp. 1517–1521.
[44] X. Zuo, X. Dai, and L. Luo, "M-SIFT: A new descriptor based on Legendre moments and SIFT," in *Proc. 3rd Int. Conf. Mach. Vis. ICMV*, 2010, pp. 183–186.
[45] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[46] J. Kautsky and J. Flusser, "Blur invariants constructed from arbitrary moments," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3606–3611, Dec. 2011.
[47] V. Ojansivu and J. Heikkilä, "A method for blur and affine invariant object recognition using phase-only bispectrum," in *Proc. Int. Conf. Image Anal. Recognit. (ICIAR)*, vol. LNCS 5112. 2008, pp. 527–536.
[48] V. Ojansivu and J. Heikkilä, "Image registration using blur-invariant phase correlation," *IEEE Signal Process. Lett.*, vol. 14, no. 7, pp. 449–452, Jul. 2007.
[49] S. Tang, Y. Wang, and Y.-W. Chen, "Blur invariant phase correlation in X-ray digital subtraction angiography," in *Proc. IEEE/CME Int. Conf. Complex Med. Eng.*, May 2007, pp. 1715–1719.
[50] M. Pedone, J. Flusser, and J. Heikkilä, "Blur invariant translational image registration for N-fold symmetric blurs," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3676–3689, Sep. 2013.
[51] M. Pedone, J. Flusser, and J. Heikkilä, "Registration of images with N-fold dihedral blur," *IEEE Trans. Image Process.*, vol. 24, no. 3, pp. 1036–1045, Mar. 2015.
[52] V. Ojansivu and J. Heikkilä, "Blur insensitive texture classification using local phase quantization," in *Proc. Congr. Image Signal Process. (CISP)*, vol. LNCS 5099. 2008, pp. 236–243.
[53] V. Ojansivu, E. Rahtu, and J. Heikkilä, "Rotation invariant local phase quantization for blur insensitive texture analysis," in *Proc. 19th Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2008, pp. 1–4.
[54] E. Rahtu, J. Heikkilä, V. Ojansivu, and T. Ahonen, "Local phase quantization for blur-insensitive image analysis," *Image Vis. Comput.*, vol. 30, no. 8, pp. 501–512, Aug. 2012.
[55] T. Ahonen, E. Rahtu, V. Ojansivu, and J. Heikkilä, "Recognition of blurred faces using local phase quantization," in *Proc. 19th Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2008, pp. 1–4.
[56] J. Flusser, T. Suk, and S. Saic, "Recognition of images degraded by linear motion blur without restoration," *Comput. Suppl.*, vol. 11, pp. 37–51, 1996.
[57] J. Flusser, T. Suk, and S. Saic, "Image features invariant with respect to blur," *Pattern Recognit.*, vol. 28, no. 11, pp. 1723–1732, Nov. 1995.
[58] J. Flusser and B. Zitová, "Invariants to convolution with circularly symmetric PSF," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2004, pp. 11–14.
[59] J. Flusser, T. Suk, J. Boldyš, and B. Zitová, "Projection operators and moment invariants to image blurring," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 786–802, Apr. 2015.
[60] J. Liu and T. Zhang, "Recognition of the blurred image by complex moment invariants," *Pattern Recognit. Lett.*, vol. 26, no. 8, pp. 1128–1138, Jun. 2005.
[61] B. Xiao, J.-F. Ma, and J.-T. Cui, "Combined blur, translation, scale and rotation invariant image recognition by Radon and pseudo-Fourier–Mellin transforms," *Pattern Recognit.*, vol. 45, no. 1, pp. 314–321, Jan. 2012.
[62] R. Gopalan, P. Turaga, and R. Chellappa, "A blur-robust descriptor with applications to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1220–1226, Jun. 2012.
[63] J. Flusser, T. Suk, and B. Zitová, *Moments and Moment Invariants in Pattern Recognition*. Chichester, U.K.: Wiley, 2009.
[64] Z. Zhang, E. Klassen, A. Srivastava, P. Turaga, and R. Chellappa, "Blurring-invariant Riemannian metrics for comparing signals and images," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Nov. 2011, pp. 1770–1775.
[65] Z. Zhang, E. Klassen, and A. Srivastava, "Gaussian blurring-invariant comparison of signals and images," *IEEE Trans. Image Process.*, vol. 22, no. 8, pp. 3145–3157, Aug. 2013.
[66] J. Kautsky, J. Flusser, B. Zitová, and S. Šimberová, "A new wavelet-based measure of image focus," *Pattern Recognit. Lett.*, vol. 23, no. 14, pp. 1785–1794, Dec. 2002.
[67] J. Flusser, "On the independence of rotation moment invariants," *Pattern Recognit.*, vol. 33, no. 9, pp. 1405–1410, Sep. 2000.
[68] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, "A statistical evaluation of recent full reference image quality assessment algorithms," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3440–3451, Nov. 2006.
[69] E. C. Larson and D. M. Chandler, "Most apparent distortion: Full-reference image quality assessment and the role of strategy," *J. Electron. Imag.*, vol. 19, no. 1, pp. 011006-1–011006-21, Jan. 2010.
[70] S. Z. Li, Z. Lei, and M. Ao, "The HFB face database for heterogeneous face biometrics research," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops (CVPR Workshops)*, Jun. 2009, pp. 1–8.
[71] D. Yi, R. Liu, R. Chu, Z. Lei, and S. Z. Li, "Face matching between near infrared and visible light images," in *Proc. Int. Conf. Adv. Biometrics (ICB)*, Aug. 2007, pp. 523–530.
[72] F. Šroubek and J. Flusser, "Multichannel blind iterative image restoration," *IEEE Trans. Image Process.*, vol. 12, no. 9, pp. 1094–1106, Sep. 2003.

Jan Flusser (M'93–SM'03) received the M.Sc. degree in mathematical engineering from Czech Technical University, Prague, Czech Republic, in 1985, the Ph.D. degree in computer science from the Czechoslovak Academy of Sciences, in 1990, and the D.Sc. degree in technical cybernetics in 2001. From 1995 to 2007, he was the Head of the Department of Image Processing. Since 1985, he has been with the Institute of Information Theory and Automation, Czech Academy of Sciences. Since 2007, he has been the Director of the Institute of Information Theory and Automation. He is currently a Full Professor of Computer Science with the Faculty of Nuclear Science and Physical Engineering, Czech Technical University, and the Faculty of Mathematics and Physics, Charles University, Prague, where he gives undergraduate and graduate courses on digital image processing, pattern recognition, and moment invariants and wavelets. He has authored or co-authored over 200 research publications in his research areas, including the monographs *Moments and Moment Invariants in Pattern Recognition* (Wiley, 2009) and *2D and 3D Image Analysis by Moments* (Wiley, 2016). His research interest covers moments and moment invariants, image registration, image fusion, multichannel blind deconvolution, and superresolution imaging. In 2007, he received the Award of the Chairman of the Czech Science Foundation for the best research project and won the Prize of the Academy of Sciences of the Czech Republic for the contribution to image fusion theory. In 2010, he also received by the SCOPUS 1000 Award. He received the Felber Medal of the Czech Technical University for excellent contribution to research and education in 2015.

Sajad Farokhi received the M.Sc. degree in mathematics from Payame Noor University, Shiraz, Iran, in 2011, and the Ph.D. degree in computer science from the Malaysia University of Technology, Johor Bahru, Malaysia, in 2014. He received the Post-Doctoral Fellowship in computer science from the Malaysia University of Technology in 2014 and the Czech Academy of Sciences in image processing from 2014 to 2015. He spent this fellowship at the Institute of Information Theory and Automation, Prague, Czech Republic. His research interests include, namely, invariants, face recognition, and near-infrared image processing.

Cyril Höschl IV, received the M.Sc. degree in computer science from the Faculty of Mathematics and Physics, Charles University, Prague, in 2010. He is currently pursuing the Ph.D. degree with the Institute of Information Theory and Automation, Prague, Czech Republic, under the supervision of J. Flusser. His research interests include invariants, moments, shape decomposition methods, mobile programming, and visualization of social interactions (sociomapping).

Tomáš Suk received the M.Sc. degree in electrical engineering from the Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic, in 1987, and the Ph.D. degree in computer science from the Czechoslovak Academy of Sciences, in 1992. Since 1991, he has been a Researcher with the Institute of Information Theory and Automation, Czech Academy of Sciences, Prague. He has authored 15 journal papers and more than 30 conference papers. He has co-authored the monographs *Moments and Moment Invariants in Pattern Recognition* (Wiley, 2009) and *2D and 3D Image Analysis by Moments* (Wiley, 2016). His research interests include digital image processing, pattern recognition, image filtering, invariant features, moment-based and point-based invariants, spatial transformations of images, and applications in remote sensing, astronomy, botany, medicine, and computer vision. In 2002, he received the Otto Wichterle Premium of the Czech Academy of Sciences for excellent young scientists.

Barbara Zitová received the M.Sc. degree in computer science and the Ph.D. degree in software systems from Charles University, Prague, Czech Republic, in 1995 and 2000, respectively. Since 1995, she has been with the Institute of Information Theory and Automation, Czech Academy of Sciences. Since 2008, she has been the Head of the Department of Image Processing. She gives undergraduate and graduate courses on digital image processing and wavelets in image processing with Czech Technical University and the Charles University. Her research interests include geometric invariants, image enhancement, image registration and image fusion, and image processing applications in cultural heritage. She has authored/co-authored over 50 research publications in these areas, including the monographs *Moments and Moment Invariants in Pattern Recognition* (Wiley, 2009), *2D and 3D Image Analysis by Moments* (Wiley, 2016), and tutorials at major conferences. In 2003, she received the Josef Hlavka Student Prize, the Otto Wichterle Premium of the Czech Academy of Sciences for excellent young scientists in 2006, and the SCOPUS 1000 Award for more than 1000 citations of a single paper in 2010.

Matteo Pedone received the M.Sc. degree in computer science from the La Sapienza University of Rome, Italy, in 2007, and the Ph.D. degree in computer science from the University of Oulu, Finland, in 2015. He is currently a Post-Doctoral Researcher with the Center for Machine Vision Research, University of Oulu. His research interests include computer vision, computational photography, invariant theory, statistical signal processing, differential geometry, and Clifford algebra.

Pattern Recognition Letters 69 (2016) 72–81

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Robust histogram-based image retrieval☆

Cyril Höschl IV, Jan Flusser*

Institute of Information Theory and Automation, Czech Academy of Sciences, Pod vodárenskou věží 4, 182 08 Praha 8, Czech Republic

ARTICLE INFO

Article history:
Received 27 March 2015
Available online 12 November 2015

Keywords:
Image retrieval
Noisy image
Histogram
Convolution
Moments
Invariants

ABSTRACT

We present a histogram-based image retrieval method which is designed specifically for noisy query images. The images are retrieved according to histogram similarity. To reach high robustness to noise, the histograms are described by newly proposed features which are insensitive to a Gaussian additive noise in the original images. The advantage of the new method is proved theoretically and demonstrated experimentally on real data.

1. Introduction

Since the appearance of the first image databases in the 80's, image retrieval has been the goal of intensive research. Early methods did not search the images themselves but utilized some kind of metadata and image annotation (tagging) to retrieve the desired images. As many large-scale databases do not contain any annotations (manual annotation is expensive and laborious while automatic tagging is still under development), content-based image retrieval (CBIR) methods have become one of the most important challenges in computer vision. By CBIR we understand methods that search a database and look for images which are the "most similar" (in a predefined metric) to a given query image. CBIR methods do not rely on a text annotation and/or other metadata but analyze the actual content of the images. Each image is described by a set of features (often hierarchical or highly compressive ones), which may reflect the image content characteristics the user prefers – colors, textures, dominant object shapes, etc. The between-image similarity is then measured by a proper (pseudo) metric in the corresponding feature space.

CBIR is a subjective task because there is no "objective" similarity measure between the images. Hence, many CBIR systems aim to retrieve images which are perceived as the most similar to the query image for a majority of users and the users feel this similarity at the first sight without a detailed exploration of the image content. This requirement, along with the need for a fast system response, has led to a frequent utilization of low-level lossy features based on image colors/graylevels. A typical example is an intensity or color histogram. It is well known that the histogram similarity is a salient property for human vision. Two images with similar histograms are mostly perceived as similar even if their actual content may be very different from each other. On the other hand, those images that have substantially different histograms are rarely rated by observers as similar. Another attractive property of the histogram is that, if normalized to the image size, it does not depend on image translation, rotation and scaling, and depends only slightly on elastic deformations. Thanks to this, one need not care about image geometry and look for geometric invariants. Simple preprocessing can also make the histogram insensitive to linear variations of the contrast and brightness of the image. Hence, the histogram established itself as a meaningful image characteristic for CBIR [7–9].

The histogram is rarely used for CBIR directly as it is basically for two reasons. The histogram is not only an inefficiently large structure (in case of color images, the RGB histogram is stored in a vector of 2^{24} integers, which may be even more than the memory requirement of the original image) but it is also redundantly detailed. It is sufficient and computationally efficient to capture only the prominent features of the histogram and suppress the insignificant details. To do so, some authors compressed the histogram from the full color range into few bins [3,4] while some others represented the histogram by its coefficients in a proper functional basis. The advantage of the latter approach is that the number of coefficients is a user-defined parameter – we may control the trade-off between a high compression on one hand and an accurate representation on the other hand. It is very natural to get inspired by a clear analogy between histogram of an image and a probability density function (pdf) of a random variable. In probability theory, the pdf is usually

☆ This paper has been recommended for acceptance by Nappi Michele.

* Corresponding author. Tel.: +42 2 6605 2357; Fax: +42 2 6641 4903.
E-mail addresses: hoschl@utia.cas.cz (C. Höschl IV), flusser@utia.cas.cz (J. Flusser).

http://dx.doi.org/10.1016/j.patrec.2015.10.012
0167-8655/

characterized by its moments, so it is worth applying the same approach in the histogram-based CBIR [6,10].

The CBIR methods based on comparing histograms are sensitive to noise in the images, regardless of the particular histogram representation. Additive noise results in a histogram smoothing, the degree of which is proportional to the amount of noise. This immediately leads to a drop of the retrieval performance because different histograms tend to be more and more similar to each other due to their smoothing. In digital photography, the noise is unavoidable. When taking a picture in low light, we use high ISO and/or long exposure. Both amplifies the background noise, which is present in any electronic system, such that the noise energy may be even higher than that of the signal. Particularly compact cameras and cell-phone cameras with small-size chips (i.e. devices which produce vast majority of photographs on Flickr, on other servers, and on personal websites) suffer from this kind of noise, along with an omnipresent thermal noise. In-built noise reduction algorithms are able to suppress the noise only slightly and perform at the expense of fine image details.

Although the noise in digital photographs is an issue we can neither avoid nor ignore, very little attention has been paid to developing noise-resistant CBIR methods. The authors of the papers on CBIR have either skipped this problem altogether or rely on denoising algorithms applied to all images before they enter the database. Such a solution, however, is not convenient or even not realistic, because the denoising inevitably introduces artifacts such as high-frequency cutoff, requires additional time, and mostly also needs a cooperation of the user in tuning the parameters. In this paper, we present an original histogram-based image retrieval method which is not only robust but totally resistant (at least theoretically) to additive Gaussian noise. The core idea of the method is a proper representation of the histogram by certain characteristics, which are not affected by the noise. We stress that the paper does *not* aim to evaluate in which tasks and for what purposes a histogram-based CBIR is appropriate. We rather show how, if it is appropriate, it should be implemented in the case of noisy database and/or noisy query images. Our method does not perform any denoising and cannot replace it in the applications where the noise should be suppressed to improve the visual quality of the image.

In the rest of the paper, we first describe the noise model we are working with and show how this noise influences the image histogram. Then we present a noise-resistant representation of the histogram and demonstrate the advantage of this representation in CBIR. In the experimental part, we compare the new method with several traditional approaches and demonstrate their advantages on a database of more than 70,000 images and 30,000 queries.

2. The noise model

As we already mentioned, we primarily consider the thermal noise and electronic background noise of consumer cameras. It is a common belief that such noise n can be modeled as a stationary additive Gaussian white noise (AGWN) with zero mean and standard deviation σ, and that the noise is not correlated with the original image f. If this assumption were true, the noise normalized histogram h_n would have a Gaussian form

$$h_n(t) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{t^2}{2\sigma^2}\right), \tag{1}$$

where t is the index of the graylevel. The histogram h_g of the noisy image $g = f + n$ would then be a convolution of the original histogram and the noise histogram

$$h_g(t) = (h_f * h_n)(t).$$

Apparently, such an ideal model can hardly be encountered in practice. Let us however demonstrate on an example that it performs a reasonable approximation of a real noise. In Fig. 4(a), we can see a clip of size 427 × 386 pixels of a real noisy image taken under low-light conditions. In order to separate f and n, we took this image repeatedly twenty-times and we estimated f by time-averaging these 20 frames (see Fig. 4(b)). This allows us to calculate all three histograms h_g, h_f, and h_n and a synthetic histogram $h_c = h_f * h_n$ (see Fig. 1 from top to bottom). We can see that the noisy picture histogram in Fig. 1(c) matches the synthetic histogram in Fig. 1(d). Additionally, in Fig. 2 we can see the normality plot of the image noise n is very close to a normal distribution. We repeated this experiment for many images with the same conclusion. Hence, we consider our noise model acceptable and use it for deriving a proper histogram representation.

3. Histogram representation resistant to image noise

In this section, we present a representation of the image histogram by descriptors which are not affected by AGWN. These descriptors are based on the statistical moments of the histogram, which is a common approach to the characterization of pdf's in probability theory. Let h be a pdf of a random variable X. Then the quantity

$$m_p^{(h)} = \int x^p h(x)dx \tag{2}$$

where $p = 0, 1, 2, \ldots$, is called *general moment* of the pdf. Clearly, $m_0 = 1$, m_1 equals the mean value and m_2 would equal the variance (if the histogram was centralized) of X. In general, the existence (finiteness) of the moments is not guaranteed, however if h is a (normalized) histogram, its support is bounded and all m_p's exist and are finite. On the other hand, any compactly-supported pdf can be exactly reconstructed from the set of all its moments.[1] In this sense moments provide a complete and non-redundant description of a pdf/histogram.

Unfortunately, the histogram moments themselves are affected by image noise. As the histogram of the noisy image is a smoothed version of the original histogram, it holds for its moments

$$m_p^{(g)} = \sum_{k=0}^{p} \binom{p}{k} m_k^{(n)} m_{p-k}^{(f)}. \tag{3}$$

This assertion can easily be proved just using the definitions of moments and of convolution. Since the noise is supposed to be Gaussian, h_n has a form of (1) and its moments are

$$m_p^{(n)} = \sigma^p (p-1)!! \tag{4}$$

for any even p. The symbol $k!!$ means a double factorial, $k!! = 1 \cdot 3 \cdot 5 \ldots k$ for odd k, and by definition $(-1)!! = 0!! = 1$. For any odd p the moment $m_p^{(n)} = 0$ due to the symmetry of the Gaussian distribution. Hence, (3) obtains the form

$$m_p^{(g)} = \sum_{k=0}^{[p/2]} \binom{p}{2k} \sigma^{2k} (2k-1)!! \cdot m_{p-2k}^{(f)}. \tag{5}$$

We can see that the moment of the noisy image histogram equals the moment of the clear image histogram plus some additional terms consisting of the moments of h_f of lower orders multiplied by a certain power of σ. For the first few moments we have

$$m_1^{(g)} = m_1^{(f)},$$
$$m_2^{(g)} = m_2^{(f)} + \sigma^2,$$
$$m_3^{(g)} = m_3^{(f)} + 3\sigma^2 m_1^{(f)},$$

[1] A more general moment problem is well known from theory of probability: can a given sequence be a set of moments of some compactly-supported function? The answer is yes if the sequence is completely monotonic.

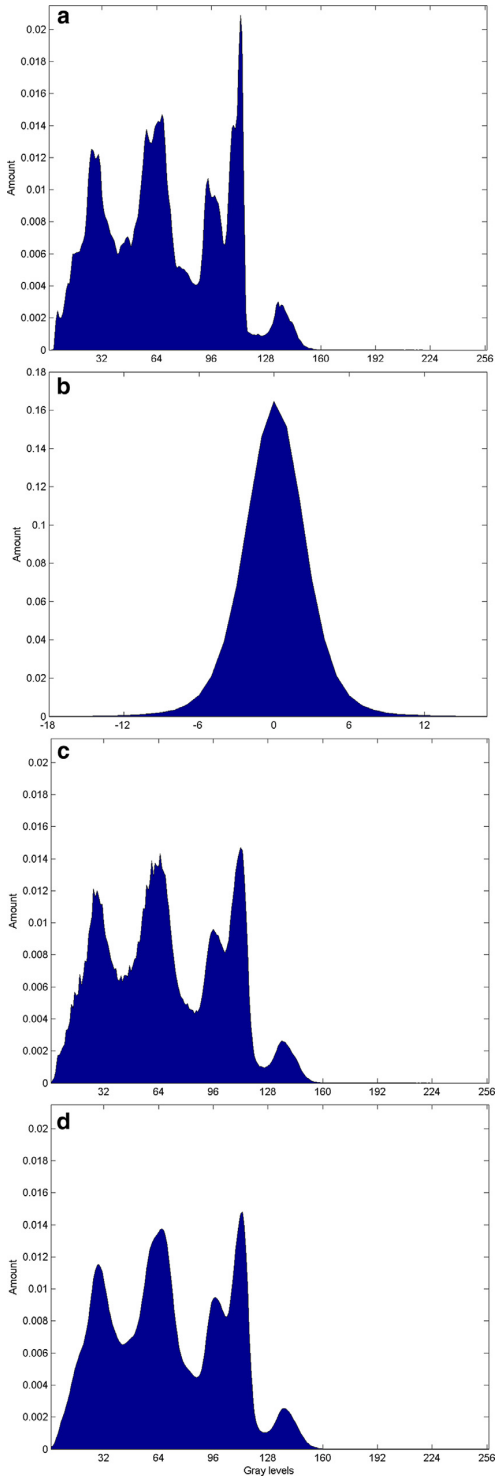


Fig. 1. Histograms of individual components of the captured image. Histogram h_f of the clear image (a), histogram h_n of the extracted image noise (note clear Gaussian shape) (b), histogram h_g of an originally captured noisy image (c), and the synthetic histogram h_c created as a convolution of the clear image histogram with the noise histogram (d). Notice the similarity of the noisy image histogram h_g and the synthetic histogram h_c.

$$m_4^{(g)} = m_4^{(f)} + 6\sigma^2 m_2^{(f)} + 3\sigma^4,$$

$$m_5^{(g)} = m_5^{(f)} + 10\sigma^2 m_3^{(f)} + 15\sigma^4 m_1^{(f)}.$$

To obtain noise-resistant descriptors, we have to eliminate the parameter σ. This can be done in a recursive manner, which leads to the definition of our histogram features

$$I_p = m_p - \sum_{k=1}^{[p/2]} (2k-1)!! \cdot \binom{p}{2k} I_{p-2k} m_2^k. \tag{6}$$

I_p can be equivalently expressed in a non-recursive form

$$I_p = \sum_{k=0}^{[p/2]} (2k-1)!! \cdot \binom{p}{2k} m_{p-2k}(-m_2)^k. \tag{7}$$

For any integer $p \geq 0$, the descriptor I_p is fully independent of the image noise regardless of the noise variance. In other words, the I_p value of an arbitrary noisy instance is the same as that of the original, and can be calculated without any denoising or estimating the noise variance (for the proof of this assertion see Appendix).

We use I_p values as histogram features for CBIR. Along with their resistance to noise, they provide an "almost complete" representation of the histogram. Having a full sequence of $I_p, \quad p = 1, 2, \ldots,$ we can recover from (7) all moments of the original histogram except $m_2^{(f)}$. This has a profound reason – since I_p is insensitive to noise, we cannot in principle recover the noise parameter σ, which influences $m_2^{(g)}$. Hence, we could recover the shape of the image histogram while its variance is a free parameter. This also corresponds to the fact that for any image $I_2 = 0$ while all other I_p's are valid. In other words, the full sequence of I_p's provides as much information about the image as its histogram itself with one degree of freedom allowing to incorporate an arbitrary unknown Gaussian smoothing of the histogram. In practice, we of course use only a finite set of these features, the number of which is determined by the user depending on the similarity of the images in the database – the more similar the images are to be discriminated, the more histogram features we need. For databases with dissimilar images, only a few (typically between 6 and 10) features are sufficient for histogram characterization, which provides an excellent compression ratio.

The intuitive meaning of the I_p's can also be understood as follows. The joint null-space of all I_p's is formed by all Gaussians, so the I_p's define the "distance" between the given histogram and the nearest Gaussian distribution. Equivalently, the I_p's actually measure the non-Gaussian component of the histogram.

It should be pointed out that the existence of such features that stay constant under a convolution of the histogram with a family of parametric kernels is a very rare phenomenon. The necessary (but not sufficient) condition is that this family must be closed with respect to convolution. In probability theory, such distributions are called *stable distributions*[2] and only three stable distributions are known in terms of elementary functions – Gaussian, Cauchy and Levy distributions.[3] Among them, only the Gaussian distribution has all finite moments, so our moment-based approach can hardly be extended to any other noise model.

It is worth mentioning that all above equations remain valid if we use *central moments* of the histogram instead of the general ones. In that way we achieve an invariance of the method to the overall brightness of the images without any histogram normalization.

[2] Equivalently, this property can be formulated such that the sum of two independent random variables, whose distributions belong to the family, has a distribution also from this parametric family.

[3] Even the generalized Gaussian distribution is not stable for exponents other than 2.

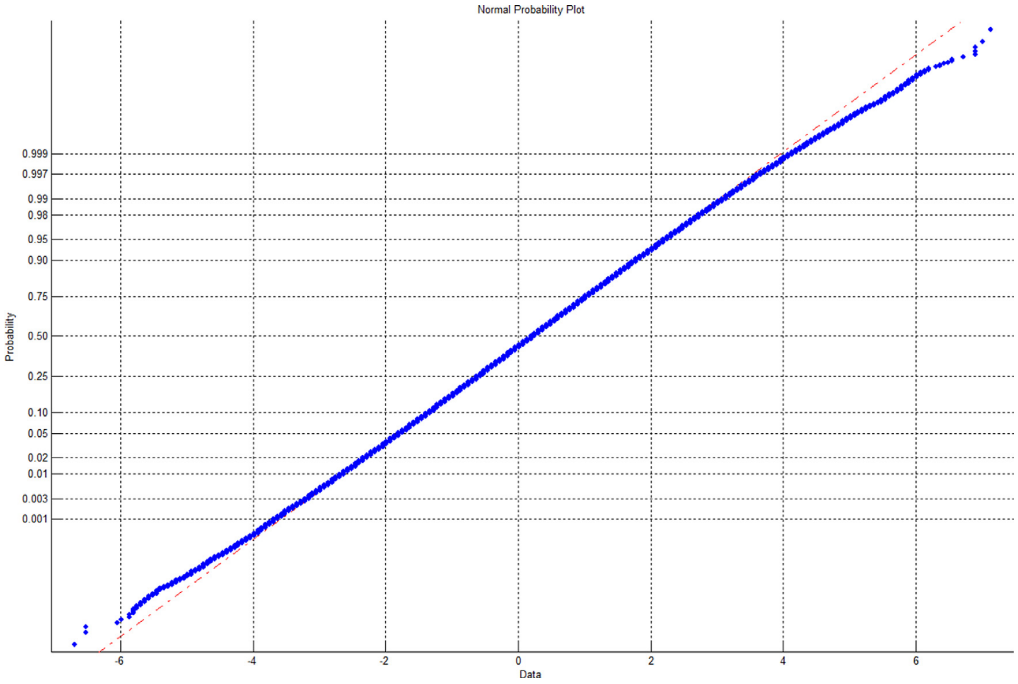


Fig. 2. Normality plot of a camera noise. Blue points indicate the noise data, the red line shows the estimated normal distribution. If the data was precisely from the normal distribution then the blue marks would be linear. Overlap of the blue marks with the dashed red line shows that the data is almost normal. The deviations on both ends are due to the value cut-off in the image. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

4. Experiments

4.1. Invariance on simulated AWGN

In the first experiment we demonstrate the invariance property of I_p (7) on pictures with simulated noise. We use a testing database of 1,000 pictures randomly gathered from Flickr[4]. The average picture size of is 1.3 Mpx and all pictures were converted to grayscale levels.

For each sample picture in the database, we created its noisy version by adding a zero-mean Gaussian white noise of various variances. It should be noted that even though the original grayscale image values range from 0 to 255, we do not cut off the values of noisy image so they can range from negative values to values higher than 255.

For each picture and each signal-to-noise ratio (SNR), we extracted two histograms: h_f of the original image and h_g of its noisy version. To show that the invariants I_p give the same results for both clear and noisy pictures, we calculated the ratio

$$r = \frac{I_p^{(f)}}{I_p^{(g)}}, \tag{8}$$

where we have applied the invariant function (7) on the histogram of the original image f divided by the invariant applied on the histogram of the noisy image g. In Fig. 3 we show the distribution of ratio r for invariants of orders $p = 3, 6, 10$ and 10 different SNRs from 5 to 32. It can clearly be observed that a majority of the ratios is almost equal to 1. It is also evident that the variance of the distribution of r increases as the SNR decreases. The fact that the ratio is not precisely 1 for all cases is because the randomly generated noise is not always exactly Gaussian. Distributions for all three chosen invariant orders are quite similar. However, the higher the order of the invariant function, the more significant the influence of the numerical errors. This can be observed as a higher variance of the distributions in the higher-order boxplots. This is an experimental verification that I_p is invariant under ideal Gaussian noise.

4.2. Invariance on real pictures

In the second experiment we demonstrate the invariance of (7) on photographs captured by a compact camera[5]. This is a much more challenging situation namely because of the value cut-offs, which violate the normality of the noise distribution.

We captured 20 different scenes under various light conditions. The light was always low to get a noticeable noise and by light changes we controlled (at least roughly) the noise variance. The estimated SNR was between 15 and 20. We took each scene 20 times and then we estimated the clear image by time-averaging, since under low light it was impossible to obtain a clear image directly (see Fig. 4 for an example).

As in the previous experiment, we evaluated the ratio (8) of invariant functions on histograms of noisy and clear pictures. To show the invariance property, the ratio r should be close to 1. Unlike the simulated noise, the real camera noise is subject to cut-off and the histogram support is bounded by the values 0–255. This causes the input data for (7) not to meet the required theoretical assumptions perfectly. In any case, the results of the invariants are quite satisfactory as we can see in Fig. 5. The median of the ratios is almost equal to 1 for all chosen invariant orders $p = 3, \ldots, 10$ and furthermore, a majority of invariant ratios is very close to 1. For a comparison and to show that this property is far from being obvious, we also calculated the same ratios for the histogram moments themselves. As one can

[4] In all our experiments we use original photographs without any postprocess modifications. Pictures are from the set used by the authors of [5].

[5] SONY Cyber-Shot DSC-H50, the resolution 3.1 Mpx was used.

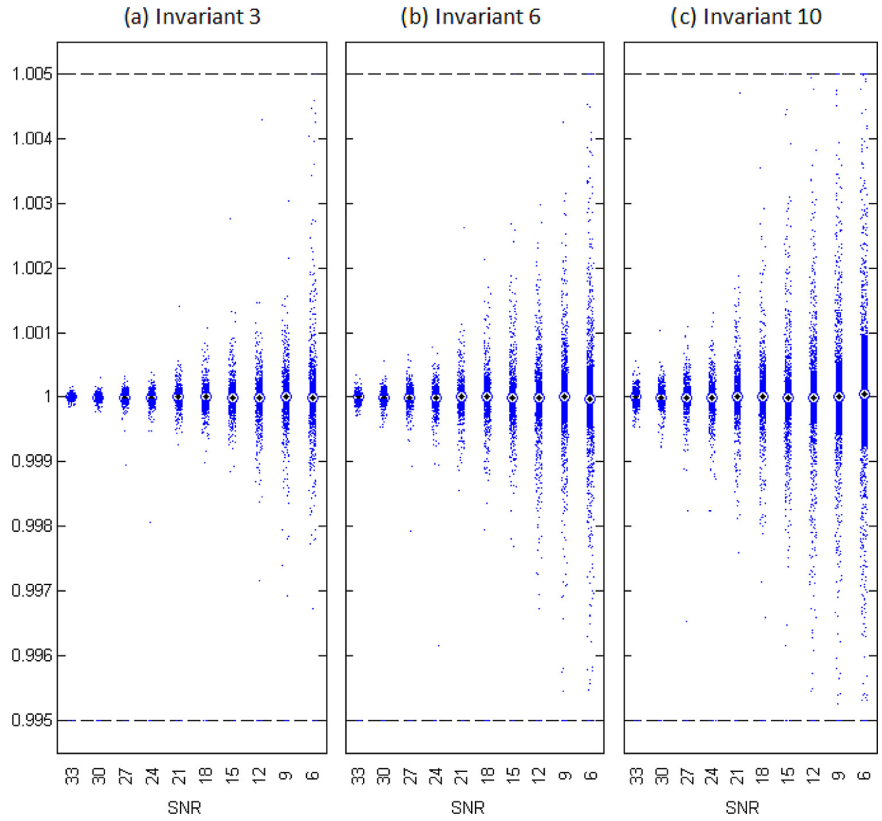


Fig. 3. The boxplots show the distribution of 1,000 ratios of invariants calculated on original images and their noisy versions. The boxplots from left to right show the results for invariant orders 3, 6 and 10 respectively. The central mark shows the median, thick bar depicts 50% of the data between 25th and 75th percentiles. Outliers outside this range are marked as dots.

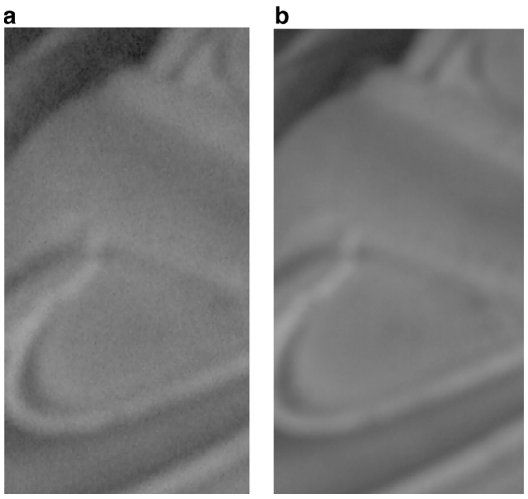


Fig. 4. A crop of the scene photographed in low light. Originally captured noisy image (a) and the noise-free image constructed by averaging 20 noisy frames of the same scene (b).

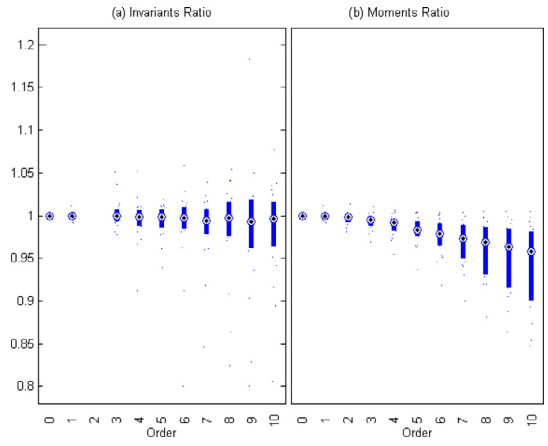


Fig. 5. (a) The boxplots show the ratio (8) of invariants calculated on histograms of real clear and noisy images. Central mark is the median of the distribution. Thick bar depicts 50% of the data between 25th and 75th percentiles. Outliers outside this range are marked as dots. (b) The boxplots show the same ratio where plain moments m_p were used instead of invariants I_p. This graph illustrates that the histogram moments cannot be used instead of the invariants since they are heavily affected by the noise.

see in Fig. 5(b), their behavior is dramatically different. The plain histogram moments are affected heavily by the noise and their relative error increases as the order grows. They do not exhibit any invariance to noise. Hence, this experiment shows that the invariants actually bring a significant added value.

4.3. Image retrieval

Content-based image retrieval is a challenging task where the user selects a query image to retrieve a list of “similar” images (the

similarity measure is pre-defined by the user, here we measure the similarity by image histograms) from a large database of pictures. The natural requirement is to avoid mismatches where the CBIR method returns images that are not related to the query image. For the human perception, two images with the same content seems similar even though one of them is affected by noise. On the other hand, CBIR methods based on comparing image histograms are sensitive to noise that modifies the histogram (see Fig. 7) and therefore standard methods may produce many mismatches. If the database system contains pictures of a similar histogram and either the input query image or the database images are affected by noise, then the danger of mismatches is high.

The aim of this experiment is to show a practical application of the proposed invariants (7) to CBIR. In this experiment the database contained clear images (or at least images with invisible noise) while the query image was always a noisy version of one database image. To make the task challenging, we intentionally included pictures of similar histograms into the database. We randomly gathered 71,842 photographs from Flickr and categorized them into 314 clusters based on histogram similarity. The image clustering was used here only to select the images for the experiments. To save time, we always limited the search to the respective cluster only. Each query image was matched only against that cluster of images with similar histograms. Matching to dissimilar histograms does not make sense because all methods correctly reject such trials. It should be noted that the similarity of the histograms may or may not correspond to the visual similarity between the images. In Fig. 6(a) we can see an example of visually different images while Fig. 6(b) provides an example of very similar images. In both cases, the histograms inside the groups are similar.

We performed 31,400 queries to achieve a statistical significance. We created query images of five SNR levels (5, 10, 15, 20 and 25). For each SNR we generated 20 different instances of the noise. The histograms of the query images were heavily smoothed due to the noise (see Fig. 7). Each query was independently answered by the following methods.

- The above mentioned invariants $I_1, \ldots, I_{10}$ calculated from the histogram of the *graylevel* image. To convert the original color images into graylevels, the Matlab function `rgb2gray` was used. The database image with the minimum Euclidean distance was retrieved. Since the invariants grow rapidly as the order p increases, we normalized the I_p's to keep them in a comparable range before calculating the distance. This method is referred to as *Invariants Gray*.
- Invariants applied on the color image channel-wise and subsequently concatenated. The feature vector was $I_1^R, \ldots, I_{10}^R, I_1^G, \ldots, I_{10}^G, I_1^B, \ldots, I_{10}^B$. The rest of the method was the same as in the previous case. This method is referred to as *RGB Vectors*.
- Invariants applied on the color image channel-wise. The concatenation was replaced by a voting scheme. The distance is calculated for each channel separately and a majority vote is applied. If at least two channels vote for the same database image, this image is retrieved. No image is retrieved if each channel votes for different database image. This method is referred to as *RGB Vote*.
- A method similar to the first one but instead of using invariants, we used plain moments $m_0, m_1, \ldots, m_{10}$ of the (graylevel) histogram. This method is referred to as *Moments Gray*.
- Full histogram matching. In this method, we match a complete graylevel histogram (256 bins) by the minimum Euclidean distance. This method is referred to as *Histogram*.
- The last method is the only one that contains denoising as a preprocessing. We denoised the query images first by a wavelet-based denoising [1] and then applied full histogram matching as in the *Histogram* method. This method is referred to as *Denoised*.

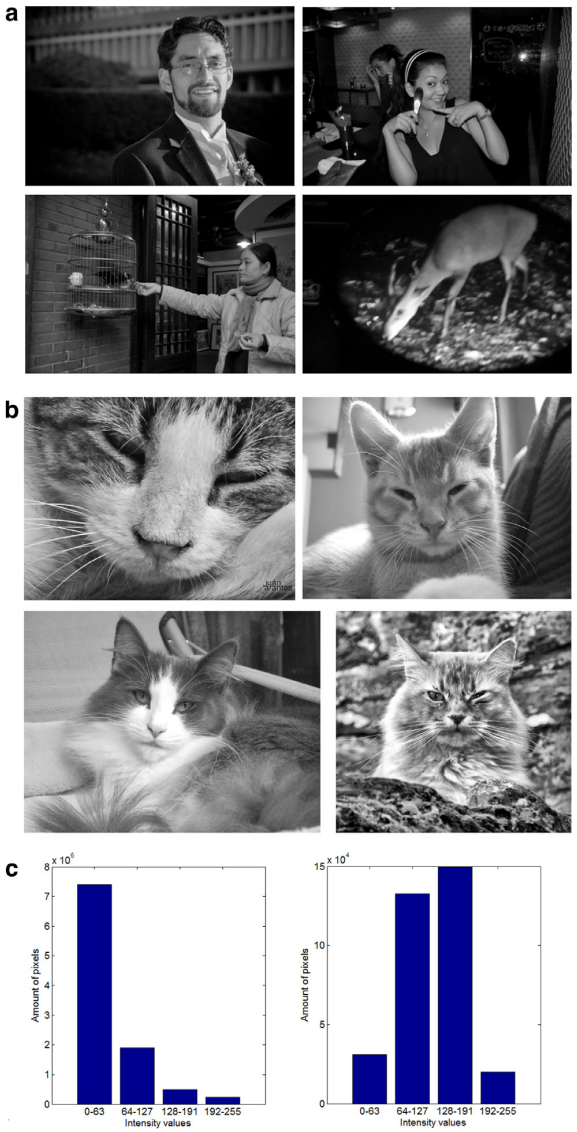


Fig. 6. Sample images from the test database. Pictures were clustered according to their histogram similarity. When considering histograms simplified into four bins, all pictures within one cluster have the same simplified histogram. In (a) and (b) there are previews of pictures from two clusters with corresponding histograms in (c) (on the left is a histogram for cluster (a) and on the right for cluster (b)). Some clusters contain pictures that have the same histogram but look differently (e.g. cluster (a)), some clusters contain pictures that look similar (e.g. cluster (b)).

Since we know the ground truth, we can evaluate the correct retrieval rate. Fig. 8 shows the results of retrieval for all the methods as a function of the SNR. The results mostly confirmed our theoretical expectation.

The *RGB Vectors* performed best, followed by the *Invariants Gray* method. The overall performance of both is very good. The *RGB Vote* performs slightly worse, which may look a bit surprising. The reason is that the majority vote from three votes is very strong criterion (we actually decide on the 2/3 majority and not on the absolute majority) and that is why we miss some correct matches. Since the *Invariants Gray* is three times faster, it may be an optimal compromise for large-scale tasks. The difference between these three

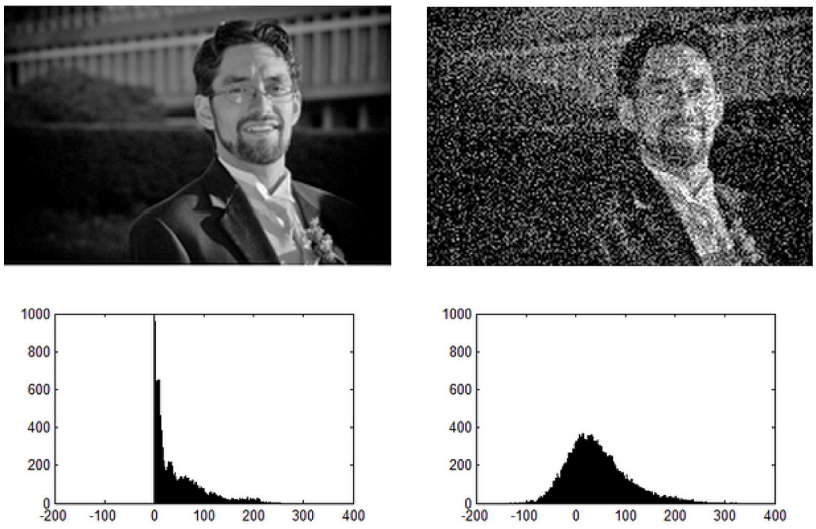


Fig. 7. Example of the query image affected by noise (SNR = 5) (right) and the clear version of the image in the database (left). On the bottom there are histograms of the images. It can be seen that the noise causes significant modification of the histogram.

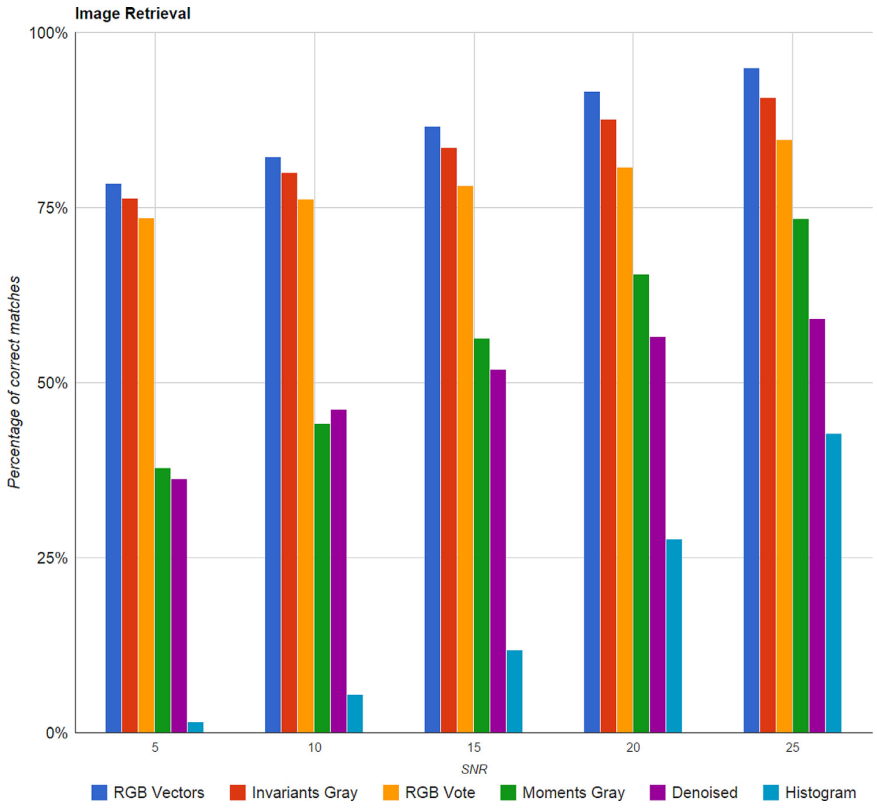


Fig. 8. Image retrieval experiment results. The graph shows percentages of correct matches for 6,280 queries for each SNR (total is 31,400 queries in 314 databases). The methods based on moment invariants outperform the methods based on plain moments or histogram matching.

methods and the other methods increases as the SNR decreases. The plain moments perform better than a complete histogram matching. The explanation is that we used only 10 low-order moments that describe global characteristics of the histogram which are less influenced by the noise than the complete histogram itself.

The most surprising result is the poor performance of the *Denoised* method. The reason is that the denoising decreases the noise level in the image but does not restore the original histogram well. It should be noted that we did not use the knowledge of the SNR when setting the parameters of the denoising algorithm. Another serious drawback of this approach is that it requires a significant extra time to perform the denoising. We also tried to replace the wavelet denoising by BM3D algorithm [2], which is one of the highest rated existing denoising methods and re-run the experiment. However, the BM3D

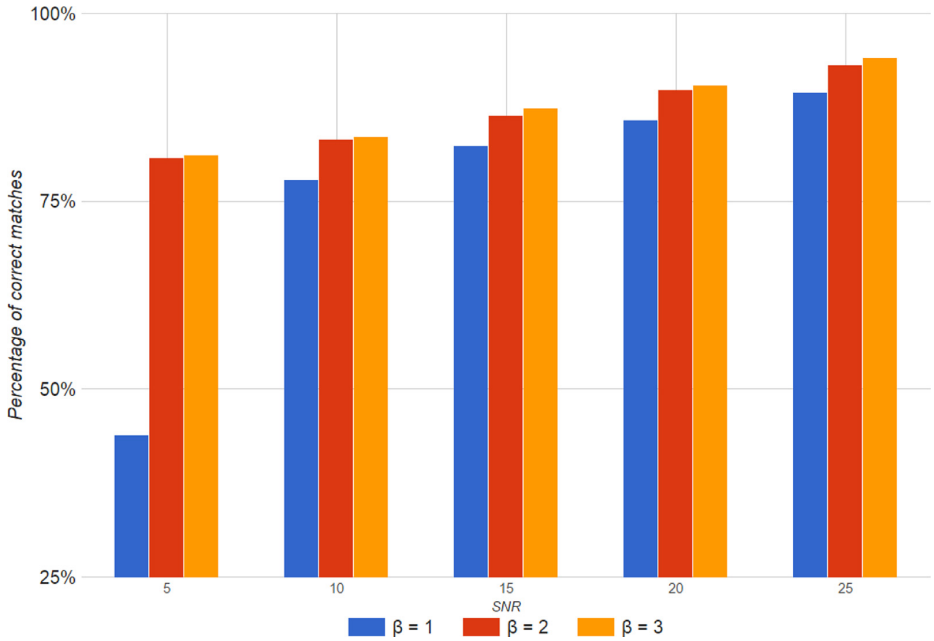


Fig. 9. Retrieval of images corrupted by heavy-tailed ($\beta = 1$), Gaussian ($\beta = 2$), and light-tailed ($\beta = 3$) noise. The graph shows percentages of correct matches by means of histogram invariants for 71,842 queries for each SNR.

Table 1
Picture database summary.

Number of databases	314
Total number of pictures	71,842
Number of queries	31,400
Average pictures count per DB	229

algorithm is so slow (10 min for one query image with 20 instances of noise) that we ran it on a small subset only to conclude that the success rate is comparable to that of the wavelet denoising. Hence, an interesting conclusion is that denoising followed by histogram matching is absolutely not suitable in terms of both success rate and speed, regardless of the particular denoising algorithm.

Finally, we tested the performance of the proposed method in the presence of noise which is not exactly Gaussian. We generated the noise underlying the *generalized normal distribution*

$$h_{\alpha,\beta}(t) = \frac{\beta}{2\alpha\Gamma(1/\beta)} \exp\left(-\frac{|t|}{\alpha}\right)^{\beta}, \tag{9}$$

This probability density function is equivalent to the Gaussian distribution for $\beta = 2$. For $\beta < 2$, it has heavier tails and for $\beta > 2$ lighter tails than the Gaussian distribution. We used two distinct β-values in this test: $\beta = 1$, which yields the Laplace distribution, and $\beta = 3$. For a comparison, we included the Gaussian case $\beta = 2$. We run the experiment with these three β-values on 71,842 test images (Table 1). We successively generated a noise of SNR ranging from 5 to 25. The retrieval results achieved by the invariants of a grey-level histogram (the same method as *Invariants Gray* in the previous experiment) are summarized in Fig. 9. One can observe two noticeable trends. The noise distribution with lighter tails does not cause any problems. The retrieval rate is fully comparable (or even slightly better in some cases) to the Gaussian noise and the method could be applied to this type of noise, too. The heavy-tailed noise significantly decreases the performance of the method on low SNR levels. As the SNR increases, the performance approaches (logically) the performance achieved for Gaussian noise.

5. Extension to color histograms

The presented invariants can be extended from 1-D graylevel histograms to color or even multispectral histograms. A complete histogram of an image with N spectral/color bands is an array of 2^{bN} integers, where b is the number of bits used to encode the pixel intensity in one band (typically $b = 8$). The size of the multispectral histogram grows exponentially with N which makes it very inefficient for small images of many bands. As the histogram size does not depend on the image size, this representation can be useful for large images with a low band number, e.g. for traditional color images with $N = 3$. Assuming a Gaussian noise is added to each band, the N-D histogram of a noisy image is again a convolution of the original histogram and the N-D Gaussian density function, which is given as

$$h_n(\mathbf{t}) = \frac{1}{\sqrt{(2\pi)^N |C|}} \exp\left(-\frac{1}{2}\mathbf{t}C^{-1}\mathbf{t}'\right), \tag{10}$$

where $\mathbf{t} \equiv (t_1, \ldots, t_N)$ and C is the noise covariance matrix.

If C is diagonal, i.e. if the noise in any two spectral bands are mutually uncorrelated, then (10) is a product of 1-D Gaussians and we can easily derive N-dimensional analogies of the invariants (6) and (7)

$$I_{\mathbf{p}} = m_{\mathbf{p}} - \sum_{\substack{\mathbf{k}=\mathbf{0} \\ \mathbf{0}<|\mathbf{k}|}}^{[\mathbf{p}/2]} (2\mathbf{k}-\mathbf{1})!! \cdot \binom{\mathbf{p}}{2\mathbf{k}} I_{\mathbf{p}-2\mathbf{k}} \mathbf{m}_2^{\mathbf{k}} \tag{11}$$

$$I_{\mathbf{p}} = \sum_{\mathbf{k}=\mathbf{0}}^{[\mathbf{p}/2]} (2\mathbf{k}-\mathbf{1})!! \cdot \binom{\mathbf{p}}{2\mathbf{k}} (-1)^{|\mathbf{k}|} m_{\mathbf{p}-2\mathbf{k}} \mathbf{m}_2^{\mathbf{k}} \tag{12}$$

where the boldface characters are used for standard vector notation and $\mathbf{m}_2 \equiv (m_{20\ldots0}, m_{02\ldots0}, \ldots, m_{00\ldots2})$.

The assumption of C being diagonal seems to be natural and it actually holds for multispectral sensors where individual bands are captured independently, such as satellite and aerial scanners, and for

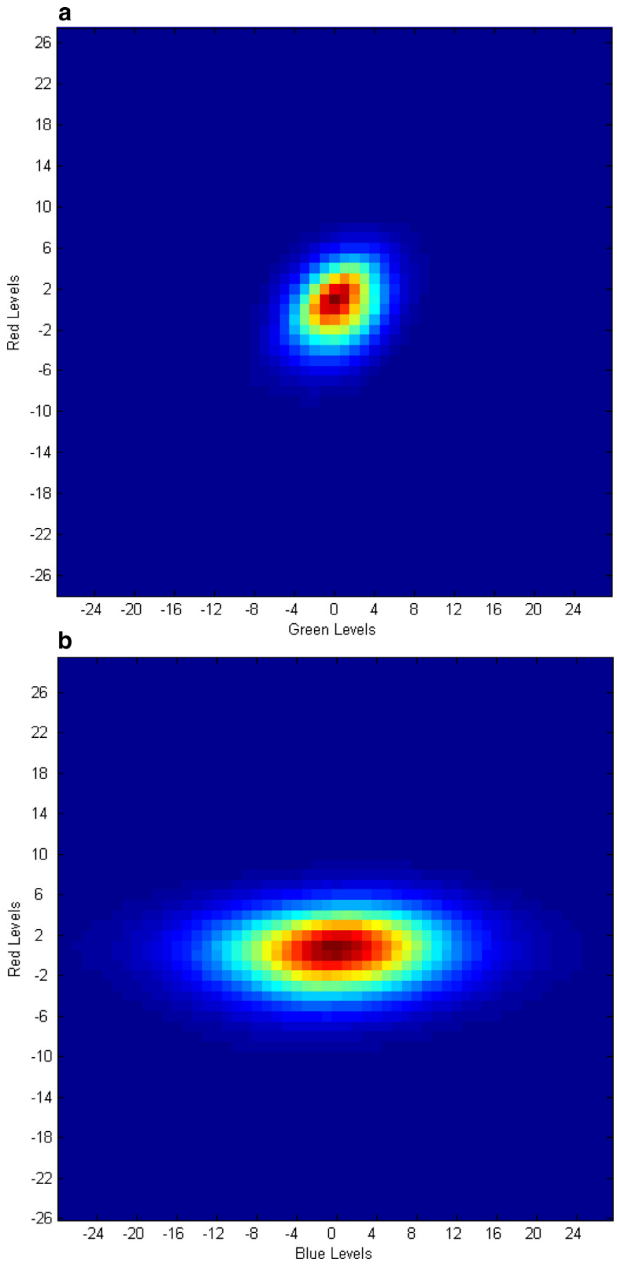


Fig. 10. An example of 2D histograms of the noise extracted from a real image. Red and green bands are correlated $\varrho = 0.33$, blue and red bands are almost independent $\varrho = 0.06$.

multimodal medical images. This is however not true for color images captured by single-chip consumer cameras, where the entering light is split into red, green and blue channels by a color filter array, most commonly arranged into Bayer pattern. To enhance the spatial resolution, the camera applies embedded interpolation algorithms on raw data (including the noise). This interpolation may introduce between-channel correlation not only of clear image data (where the correlation is expected anyway) but also of the noise components which theoretically should be independent. To illustrate this phenomenon, we extracted the noise components of a real color image using the same technique as described in Section 2 and visualized 2-D histograms of the noise (see Fig. 10). While the blue and the red bands are uncorrelated, the correlation between red and green is about $\varrho = 0.33$. We performed this test on several images with basically the same results (the particular values of course depend on the camera type, on the setting and on other conditions). Note also that the noise variances in individual channels typically differ from each other, but this is not a serious problem.

If C is not diagonal, it would still be possible to derive histogram features insensitive to such kinds of correlated noise, assuming that the eigenvectors of C are of a known orientation, which is constant for all images in question. We could rotate the histogram such that the noise becomes uncorrelated, which is always possible, and then proceed as described above. This is, however, not the case of real color noise, where the eigenvectors of C are basically random. Under such conditions, the invariant approach cannot be used correctly and we are limited to channel-wise histograms.

6. Conclusion

Histogram of a noisy image, both visual appearance and common numerical characteristics, are significantly affected by additive noise in the image. Provided the noise is Gaussian, we proposed original histogram descriptors which are invariant w.r.t. the noise. We proved that along with the theoretical invariance the descriptors are sufficiently robust on real images corrupted by thermal and electronic sensor noise. As demonstrated experimentally, the proposed descriptors can be used as the features in a histogram-based retrieval if the database and/or query images are heavily noisy and standard descriptors fail. We approved that the retrieval based on the new invariants significantly outperform the other more traditional methods included in our tests. We also proved that the method can be used even if the noise distribution is not exactly Gaussian, but has lighter tails.

Acknowledgment

This work has been supported by the Grant no. GA15-16928S of the Czech Science Foundation.

Appendix

In this Appendix we present a formal proof that the image features defined in Eq. (6) do not change under Gaussian white noise. To prove this, it is sufficient to show that they do not change if the image histogram is convolved by an arbitrary zero-mean Gaussian pdf (1) of an unknown parameter σ. We prove this by induction over p. The validity is trivial for $p = 0, 1, 2$ and can be verified easily for $p = 3$ by substitution of (5) into (6). Let us now prove (6) for an arbitrary $p > 3$ provided that it holds for all lower indices.

$$I_p^{(g)} = m_p^{(g)} - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} I_{p-2k}^{(g)} (m_2^{(g)})^k$$

where $K = [p/2]$. Using (5) and the assumption that $I_{p-2k}^{(g)} = I_{p-2k}^{(f)}$ we get

$$\begin{aligned} I_p^{(g)} &= \sum_{k=0}^{K} \binom{p}{2k} \sigma^{2k} (2k-1)!! \cdot m_{p-2k} \\ &\quad - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} I_{p-2k} (m_2 + \sigma^2)^k \\ &= m_p - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} \\ &\quad \times \left(I_{p-2k} \sum_{j=0}^{k} \binom{k}{j} \sigma^{2j} m_2^{k-j} - \sigma^{2k} m_{p-2k} \right) \end{aligned}$$

$$
\begin{aligned}
&= I_p^{(f)} - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} \\
&\quad \times \left(I_{p-2k} \sum_{j=1}^{k} \binom{k}{j} \sigma^{2j} m_2^{k-j} - \sigma^{2k} m_{p-2k} \right) \\
&= I_p^{(f)} - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} I_{p-2k} \sum_{j=1}^{k} \binom{k}{j} \sigma^{2j} m_2^{k-j} \\
&\quad + \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} \sigma^{2k} m_{p-2k} \\
&= I_p^{(f)} - A_p + B_p.
\end{aligned}
$$

In the above expressions, we dropped the index $^{(f)}$ for the sake of simplicity whenever it is clear from the context. Now we show that $A_p = B_p$, which will complete the proof. To express the double factorial, we use the relation $(2k-1)!! = (2k)!/2^k k!$.

$$
\begin{aligned}
A_p &= \sum_{k=1}^{K} \sum_{j=1}^{k} (2k-1)!! \cdot \binom{p}{2k} \binom{k}{j} I_{p-2k} \sigma^{2j} m_2^{k-j} \\
&= \sum_{j=1}^{K} \sum_{k=j}^{K} (2k-1)!! \cdot \binom{p}{2k} \binom{k}{j} I_{p-2k} \sigma^{2j} m_2^{k-j} \\
&= \sum_{j=1}^{K} \sum_{k=0}^{K-j} (2(k+j)-1)!! \cdot \binom{p}{2(k+j)} \binom{k+j}{j} I_{p-2(k+j)} \sigma^{2j} m_2^{k} \\
&= \sum_{j=1}^{K} \frac{p! \sigma^{2j}}{j! 2^j} \sum_{k=0}^{K-j} \frac{m_2^k}{k!(p-2j-2k)! 2^k} I_{p-2j-2k} \\
&= \sum_{j=1}^{K} \frac{p! \sigma^{2j}}{j! 2^j} \left(\sum_{k=1}^{K-j} \frac{m_2^k}{k!(p-2j-2k)! 2^k} I_{p-2j-2k} + \frac{I_{p-2j}}{(p-2j)!} \right)
\end{aligned}
$$

The inner sum equals, according to (6), to

$$\frac{m_{p-2j} - I_{p-2j}}{(p-2j)!}.$$

Hence,

$$A_p = \sum_{j=1}^{K} \frac{p! \sigma^{2j} m_{p-2j}}{j!(p-2j)! 2^j} = B_p. \qquad \square$$

In a similar way, by means of induction over p, it is also possible to prove the equivalence between (6) and (7). We briefly show the induction step.

$$
\begin{aligned}
I_p &= m_p - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} m_2^k I_{p-2k} \\
&= m_p - \sum_{k=1}^{K} (2k-1)!! \cdot \binom{p}{2k} m_2^k \\
&\quad \times \sum_{j=0}^{K-k} (2j-1)!! \cdot \binom{p-2k}{2j} (-m_2)^j m_{p-2k-2j} \\
&= m_p - \sum_{k=1}^{K} \sum_{j=0}^{K-k} (-1)^j \frac{p!}{2^{k+j} k! j! (p-2k-2j)!} m_2^{k+j} m_{p-2k-2j} \\
&= m_p - \sum_{k=1}^{K} \sum_{j=k}^{K} (-1)^{j-k} \frac{p!}{2^j k! (j-k)! (p-2j)!} m_2^j m_{p-2j} \\
&= m_p - \sum_{j=1}^{K} \sum_{k=1}^{j} (-1)^{j-k} \frac{p!}{2^j k! (j-k)! (p-2j)!} m_2^j m_{p-2j} \\
&= m_p - \sum_{j=1}^{K} (-1)^j \frac{p!}{2^j (p-2j)!} m_2^j m_{p-2j} \sum_{k=1}^{j} \frac{(-1)^k}{k!(j-k)!}
\end{aligned}
$$

Since

$$\sum_{k=1}^{j} (-1)^k \cdot \binom{j}{k} = -1$$

for any j, we obtain

$$
\begin{aligned}
I_p &= m_p + \sum_{j=1}^{K} (2j-1)!! \binom{p}{2j} (-m_2)^j m_{p-2j} \\
&= \sum_{j=0}^{K} (2j-1)!! \binom{p}{2j} (-m_2)^j m_{p-2j},
\end{aligned}
$$

which exactly matches Eq. (7).

References

[1] R. Coifman, D. Donoho, Translation-invariant de-noising, in: A. Antoniadis, G. Oppenheim (Eds.), Wavelets and Statistics, Lecture Notes in Statistics, vol. 103, Springer, New York, 1995, pp. 125–150, doi:10.1007/978-1-4612-2544-7_9.

[2] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, Image denoising by sparse 3-D transform-domain collaborative filtering, IEEE Trans. Image Process. 16 (8) (2007) 2080–2095, doi:10.1109/TIP.2007.901238.

[3] S. Jeong, Histogram-Based Color Image Retrieval., Technical Report, Stanford University, 2001. Psych221/EE362 Project Report.

[4] G.-H. Liu, L. Zhang, Y.-K. Hou, Z.-Y. Li, J.-Y. Yang, Image retrieval based on multitexton histogram, Pattern Recognit. 43 (7) (2010) 2380–2389, doi:10.1016/j.patcog.2010.02.012.

[5] B. Mahdian, R. Nedbal, S. Saic, Blind verification of digital image originality: A statistical approach, IEEE Trans. Inf. Forensics Secur. 8 (9) (2013) 1531–1540, doi:10.1109/TIFS.2013.2276000.

[6] M. Mandal, T. Aboulnasr, S. Panchanathan, Image indexing using moments and wavelets, IEEE Trans. Consum. Electron. 42 (1996) 557–565.

[7] G. Pass, R. Zabih, Histogram refinement for content-based image retrieval, in: Proceedings of the Third IEEE Workshop on Applications of Computer Vision (WACV), 1996. '96., 1996, pp. 96–102, doi:10.1109/ACV.1996.572008.

[8] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comput. Vis. 7 (1) (1991) 11–32, doi:10.1007/BF00130487.

[9] W. Xiaoling, A novel circular ring histogram for content-based image retrieval, in: Proceedings of the First International Workshop on Education Technology and Computer Science (ETCS), 2009, vol. 2, 2009, pp. 785–788, doi:10.1109/ETCS.2009.437.

[10] P.-T. Yap, R. Paramesran, Content-based image retrieval using Legendre chromaticity distribution moments, IEE Proc. Vis. Image Signal Process. 153 (1) (2006) 17–24, doi:10.1049/ip-vis:20045064.