Interactive Analysis of Multispectral and Hyperspectral Image Data

Interaktive Analyse von multispektralen und hyperspektralen Bilddaten

Der Technischen Fakultät
der Friedrich-Alexander-Universität
Erlangen-Nürnberg

zur

Erlangung des Doktorgrades Dr.-Ing.

vorgelegt von

Johannes Michael Jordan
aus Nürnberg

Als Dissertation genehmigt
von der Technischen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der mündlichen Prüfung:	15.05.2017
Vorsitzender des Promotionsorgans:	Prof. Dr.-Ing. R. Lerch
Gutachter:	Prof. Dr.-Ing. J. Hornegger
	Prof. Dr.-Ing. M. Stamminger

Abstract

A multispectral or hyperspectral sensor captures images of high spectral resolution by dividing the light spectrum into many narrow bands. With the advent of affordable and flexible sensors, the modality is constantly widening its range of applications. This necessitates novel tools that allow general and intuitive analysis of the image data. In this work, a software framework is presented that bundles interactive visualization techniques with powerful analysis capabilities and is accessible through efficient computation and an intuitive user interface. Towards this goal, several algorithmic solutions to open problems are presented in the fields of edge detection, clustering, supervised segmentation and visualization of hyperspectral images.

In edge detection, the structure of a scene can be extracted by finding discontinuities between image regions. The high dimensionality of hyperspectral data poses specific challenges for this task. A solution is proposed based on a data-driven pseudometric. The pseudometric is computed through a fast manifold learning technique and outperforms established metrics and similarity measures in several edge detection scenarios.

Another approach to scene understanding in the hyperspectral or a derived feature space is data clustering. Through pixel-cluster assignment, a global segmentation of an image is obtained based on reflectance effects and materials in the scene. An established mode-seeking method provides high-quality clustering results, but is slow to compute in the hyperspectral domain. Two methods of speedup are proposed that allow computations for interactive use. A further method is proposed that finds clusters in a learned topological representation of the data manifold. Experimental results demonstrate a quick and accurate clustering of the image data without any assumptions or prior knowledge, and that the proposed methods are applicable for the extraction of material prototypes and for fuzzy clustering of reflectance effects.

For supervised image analysis, an algorithm for seed-based segmentation is introduced to the hyperspectral domain. Specific segmentations can be quickly obtained by giving cues about regions to be included in or excluded from a segment. The proposed method builds on established similarity measures and the proposed data-driven pseudometric. A new benchmark is introduced to assess its performance.

The aforementioned analysis methods are then combined with capable visualization techniques. A method for non-linear false-color visualization is proposed that distinguishes captured spectra in the spatial layout of the image. This facilitates the finding of relationships between objects and materials in the scene. Additionally, a visualization for the spectral distribution of an image is proposed. Raw data exploration becomes more feasible through manipulation of this plot and its link to traditional displays. The combination of false-color coding, spectral distribution plots, and traditional visualization enables a new workflow in manual hyperspectral image analysis.

Zusammenfassung

Multispektrale und hyperspektrale Kameras nehmen Bilder mit hoher spektraler Auflösung auf, indem das Lichtspektrum in viele schmale Bänder zerlegt wird. Durch die Verfügbarkeit von günstigen und flexiblen Bildsensoren wird die Technologie für stetig neue Anwendungen interessant. Dabei entsteht ein Bedarf für neue Werkzeuge, die eine allgemeine und intuitive Analyse der Bilder erlauben. Diese Arbeit führt ein Softwareframework ein, das interaktive Visualisierungstechniken mit performanten Analysefähigkeiten kombiniert und durch eine effiziente und intuitive graphische Oberfläche zugänglich ist. Hierfür werden algorithmische Lösungen zu bisher offenen Problemen in den Bereichen Kantendetektion, Clustering, nutzergestützte Segmentierung und Visualisierung von hyperspektralen Bildern entwickelt.

Kantendetektion ermöglicht durch das Finden von Unstetigkeiten zwischen Bildbereichen, Rückschlüsse auf die Struktur der Bildszene zu ziehen. Dabei bringt die hohe spektrale Auflösung bestehende Verfahren an und über ihre Grenzen. Um diese zu verschieben, stellen wir eine datenbezogene Halbmetrik vor, die mithilfe einer Methode zum schnellen Erlernen einer Mannigfaltigkeit berechnet wird. Sie erzielt gegenüber etablierten Metriken und Ähnlichkeitsmaßen deutliche Verbesserungen in verschiedenen Kantendetektionsszenarien.

Die Bildszene kann auch im hyperspektralen Raum oder einem davon abgeleiteten Datenraum mittels Datenclustering greifbar werden. Durch die Clusterzugehörigkeiten der Pixel erhält man eine globale Segmentierung des Bildes basierend auf Reflexionseffekten und Materialien in der Szene. Eine etablierte Clustering-Methode, die sehr gute Ergebnisse liefert, ist im hyperspektralen Raum nur langsam zu berechnen. Zwei Methoden zur Beschleunigung werden vorgestellt, die diese Berechnung im interaktiven Zeitrahmen ermöglichen. Eine weitere Methode wird vorgeschlagen, die eine erlernte topologische Repräsentation der von den Daten aufgespannten Mannigfaltigkeit zum Clustering nutzt. Die experimentellen Ergebnisse zeigen, dass ein schnelles und präzises Clustering der Bilddaten ohne spezifische Annahmen oder weiteres Vorwissen erreicht werden kann. Ferner wird die Extraktion von Materialprototypen sowie weiches Clustern von Reflexionseffekten ermöglicht.

Um die nutzergestützte Bildanalyse zu erlauben, führen wir einen Algorithmus zur saatbasierten Segmentierung für hyperspektrale Bilder ein. Er verwendet Hinweise über Regionen, die im Segment enthalten oder von ihm ausgeschlossen sind. So kann der Nutzer rasch die gewünschte Segmentierung erzielen. Die vorgestellte Methode nutzt etablierte Ähnlichkeitsmaße sowie die neue datenbezogene Halbmetrik. Ihre Tauglichkeit wird mittels eines neu erstellten Benchmarks nachgewiesen.

Schließlich werden die genannten Analysemethoden mit leistungsfähigen Visualisierungstechniken kombiniert. Eine Methode zur nichtlinearen Falschfarbendarstellung wird eingeführt, die aufgenommene Spektren in der Anordnung des Bildes differenziert und es ermöglicht, Zusammenhänge zwischen den Objekten und Materialien in der Szene zu finden. Ebenso wird eine Visualisierung der spektralen Verteilung eines Bildes eingeführt. Interaktive Manipulation dieses Plots und seine Verknüpfung mit anderen Ansichten ermöglicht die Erkundung unverarbeiteter Daten. Die Kombination von Falschfarbendarstellung, Plots der spektralen Verteilung und traditioneller Visualisierungen erlaubt einen neuen Arbeitsablauf bei der manuellen Inspektion von multispektralen und hyperspektralen Bildern.

Acknowledgment

A number of people besides myself are responsible for the event and final shape of this thesis. Joachim Hornegger invited me to work at the Pattern Recognition Lab and never stopped to show his confidence in me as a person and in my ability to perform good research work. I could always count on his support, not only in research. My advisor in Erlangen was Elli Angelopoulou, who introduced me to multispectral imaging, gave me a lot of freedom in the directions of research, and also got me in contact with Antonio Robles-Kelly. My research visit of Antonio's group and his supervision during that time exposed me to several new, important aspects of research and this research field in particular.

Marc Stamminger is reviewing this work. He is the driving force behind the research training group "Heterogeneous Image Systems", which had a great impact on my PhD program, including the outstanding opportunity to visit the group of Antonio in Canberra.

My colleagues at the Pattern Recognition Lab provided for the work environment you would wish for and always had a helping hand for me. In particular, I would like to thank Christian Rieß, Eva Eibenberg, Martin Berger, Andre Linarth, Vincent Christlein, David Bernecker and Hannes Hofmann. The following students were involved in various stages of my research and work on the software, and greatly contributed to it: Georg Altmann, Daniel Danner, Tobias Würfl, Petr Koupý, Stefan Ploner, Ralph Müssig, Aleksander Cieślak, and Michal Cieślak.

During the final months of this thesis, I was invaluably supported by Christian Rieß and Andreas Maier. I also received technological assistance from Raisa Kociurzynski and Maya Angelova.

Some of the core ideas in this thesis were inspired by the teachings of Volker Strehl and the late Gabriella Kókai, and a talk by the late Dirk Bartz.

Johannes Jordan

Contents

Chapter 1

Introduction

In our daily lives we solve many challenging vision problems, most of the time even without giving them a further thought. Detection, recognition, tracking are classical computer vision problems where human performance often stays unmatched by algorithmic solutions to date. On the other hand, autonomous systems strive to beat human performance in many fields, e.g. to improve traffic safety with driving assistance systems; even heading towards completely autonomously controlled vehicles. One corner stone to enable machine vision to beat human vision are sensors that can see what a human eye cannot.

A multispectral or hyperspectral sensor is such a sensor. It allows to capture rich reflectance information that is not accessible to our own visual system. While the latter is based on three stimuli – different cones in the eye that perceive the colors red, green, and blue –, these sensors provide a much better discrimination in the visible spectrum, or even beyond. Multispectral and hyperspectral imaging have played a key role in the field of remote sensing for decades, ever since reconnaissance aircraft and satellites became equipped with such sensor systems for earth observation. Today, the range of available sensor designs and applications for multispectral and hyperspectral imaging has widened significantly, far beyond remote sensing and astronomy, for example in areas like cultural heritage, agriculture, and medical imaging.

1.1 Motivation

In a multispectral image, each pixel is a high-dimensional vector of intensity values, where each value corresponds to the scene radiance over a small range of wavelengths (a narrow band). The high-dimensional nature of the data and its strong inter-band correlations pose challenges to computer vision algorithms. It turns out a superior sensor is only half the battle: Software adaptations are needed to expose and exploit the information contained in the data.

A similar problem arises when one intends to manually process such high-dimensional data. How do you view a hyperspectral image? High-dimensional data is unintuitive, and it typically cannot be fully represented in a single, static display. Many new applications bring this imaging domain to new users – domain experts

who did not work with hyperspectral data before but need to find and analyze details that are seemingly hidden in the data. The typical answer to these problems is data reduction. However, when is data reduction really needed and if so, can we find new ways to achieve it?

One major goal of this thesis is to provide new answers to both parts of this question. Where traditional displays are based on static imagery and dimensionality reduction, we investigate a new workflow for interactive inspection that works with simultaneous visualization of different aspects of the information; including the display of all raw spectra contained in the image or a region-of-interest. To help unravel the rich data, we develop new tools for the user to separate parts of the data automatically or with a guided algorithm that was adapted to hyperspectral images. We also improve on the computational performance of algorithms that already work on hyperspectral input to enable their use in the interactive analysis. One important ingredient of these efforts is reducing data in several ways, based on a fast-to-compute manifold learning method.

1.2 Contributions

In this thesis, we introduce a novel software platform for interactive analysis and visualization of multispectral and hyperspectral data to help proliferate research and application of multispectral and hyperspectral imaging [Jord 16b]. Towards this goal, the following work was carried out:

- A probabilistic manifold learning method [Jord 14, Robl 15],
- a new pseudometric for edge detection [Jord 11],
- fast global clustering with superpixel support [Jord 13b],
- supervised segmentation for hyperspectral images [Jord 12b],
- fast non-linear false-color visualization [Jord 13a], and
- efficient, interactive visualization of spectral distributions [Jord 10].

The *Gerbil* hyperspectral image visualization and analysis framework originating from this work is available as free software for use in teaching, research and industry [Jord 16a]. It incorporates the algorithms presented in this thesis with the state-of-the-art in interactive hyperspectral image analysis.

1.3 Thesis Outline

The thesis is organized as follows. Chapter 2 introduces the multispectral and hyperspectral imaging modalities. Due to a new software platform being an integral part of this thesis, currently available software for hyperspectral image analysis is reviewed. As a foundation for the algorithms presented in this thesis, it is explained

how spectral image data is typically described and how spectra are compared. We also mention the challenges inherent to this domain.

In Chapter 3, we discuss the prevalent topic of dimensionality reduction in hyperspectral image processing. Dimensionality is often reduced as a feature extraction step to combat the problems inherent to the high dimensionality of the captured data. After an overview on well-studied transformations typically employed in this context, we concentrate on the Self-organizing Map (SOM), which achieves vector quantization and dimensionality reduction through manifold learning. In our work, we found new variants of the SOM to aid in a range of traditional image processing problems for hyperspectral data [Jord 11, Jord 13a], which are introduced at this point. We follow up on this by discussing a new point of view on how the mechanisms inherent to the SOM can be cast as a probabilistic manifold learning algorithm [Jord 14], a joint work with Antonio Robles-Kelly. The chapter is concluded by a range of experiments seeking to validate both the viability of the SOM for manifold learning on hyperspectral images, and the performance improvements achieved by our own contributions to the algorithm. For the latter, we derive a new method for SOM-based classification of spectra.

Chapter 4 visits three prominent image processing problems in the context of hyperspectral image analysis. First, the problem of edge detection is tackled, where points in the image are identified that observe spectral discontinuities. The performance of current state-of-the art methods in hyperspectral edge detection is discussed. Then, we propose two variants of an algorithm that defines a dependable and globally consistent measure of spectral discontinuity. The first variant is based on the traditional SOM and was published in [Jord 11]. The second variant leverages our improved SOM algorithm introduced in Chapter 3. We compare the performance of these two algorithms with state-of-the-art edge detection methods.

Second comes global clustering. Here, we briefly discuss popular directions in image clustering, before going in more detail for the family of so-called shift-based methods. The Mean Shift algorithm is a prominent example and finds clusters in the data by density gradient estimation. We show that a fast variant of mean shift, the Fast Adaptive Mean Shift (FAMS), can provide good quality segmentations of hyperspectral images. To obtain such segmentations under interactive time constraints, we develop two variants of FAMS based on a sparsification or quantization of the data distribution. One is based on superpixel segmentation [Jord 13b], the other on manifold learning [Jord 14]. A third proposed method works with the topology derived by a SOM. We assess the utility of our methods by comparing clustering results with FAMS and another algorithm designed for hyperspectral data, and show an application in fuzzy clustering.

The third problem we visit is supervised segmentation. Other than global clustering methods, supervised segmentation finds locally connected segments and is guided by the user. While supervised segmentation can be very helpful for interactive analysis, to the best of our knowledge, it was not studied so far for hyperspectral images. We present an efficient solution to this problem that works with either established measures of spectral dissimilarity or the new measure introduced at the beginning of this chapter [Jord 12b].

Chapter 5 complements the analysis carried out in Chapter 4 by the visualization of multispectral and hyperspectral image data, which is crucial for interactive inspection. First, traditional display methods are discussed before we introduce two new approaches. Our first approach follows the traditional false-color technique. Here, a color image is produced that follows the spatial layout of the high-dimensional image, but derives an artificial coloring from computed data. We generate such a coloring using our own variant of the SOM that was introduced in Chapter 3 [Jord 13a]. Our second approach does not work in the spatial layout of the image, but rather visualizes the data by the means of an interactive plot of the image's underlying spectral distribution [Jord 10]. The discussion includes further refinements to the plotting method.

Based on this work, we obtain a new workflow in visual hyperspectral image analysis that tightly integrates traditional displays, false-coloring and interactive spectral plots with unsupervised and supervised analysis methods [Jord 16b]. We illustrate this workflow within our software framework. In the experimental validation of our methods, we contrast them with existing approaches, but also reveal relations between them and how the spectral distribution plots can be used to compare feature spaces and evaluate other algorithmic results.

To conclude, Chapter 6 provides an outlook on prolific further research directions before we summarize the findings in this thesis in Chapter 7.

Chapter 2

Background

While having close algorithmic ties to other disciplines, imaging spectroscopy, or hyperspectral imaging, resides in its own realm in image processing. In this chapter, we first give an overview of sensors, applications, and available image data that is widely used for evaluation. A comprehensive study is provided on existing software frameworks that address the need for hyperspectral analysis tools. Then, we review how multispectral and hyperspectral image data is typically described internally, and which measures are used to differentiate between individual spectra. Finally, we look at the challenges that are present today for intuitive inspection and analysis of hyperspectral data.

2.1 Imaging Spectroscopy

The term imaging spectroscopy describes the measurement and examination of the light spectrum, as opposed to monochromatic measurements, on an image level. In practice, each pixel of the image contains a considerably high-resolution representation of the incident light spectrum. The benefits of such an instrument when compared to a regular color sensor can be easily illustrated by an example. A regular color sensor separates the incoming visible light spectrum into three wide bands, representing red, green, and blue (RGB) each. While this may generally suffice for our own image viewing pleasure, it is a very limited view when going into specific color-based vision applications. For example, the absorption spectra of the components present in the human skin layers are depicted in Figure 2.1 [Mals 11]. Particularly characteristic is the w-shape between wavelengths of 500 nm and 600 nm formed by oxygenated hemoglobin (oxyhemoglobin). Concentration of oxyhemoglobin is a feature used for measuring oxygen saturation, which by itself is an important indicator in medical diagnosis. With this knowledge, we are not only able to visually recognize human skin with overwhelming reliability, we can also derive dermatological properties by decomposing the absorbers, to diagnose bruises, the degree of burn wounds, diabetes, and skin cancer [Lu 14]. Note that in the considerably lower spectral resolution of RGB, this feature cannot be observed.

Several instrument designs exist to obtain a representation of the incident light with moderate to ultra-high spectral resolutions to facilitate this application and

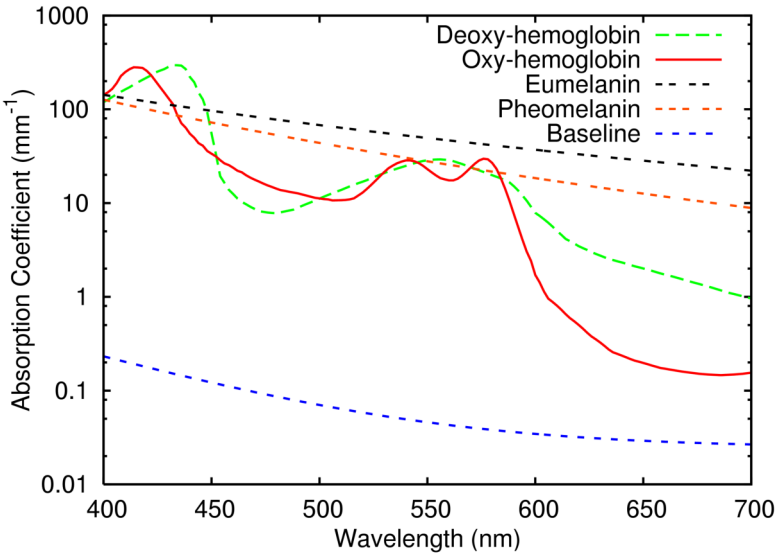


Figure 2.1: Absorption spectra of different absorbers in skin [Mals 11]. Baseline absorption stems from structures like organelles and cell membranes. Courtesy of Eva Eibenberger.

many others. The distinction between *multispectral* and *hyperspectral* instruments is diffuse. Per-se, multispectral describes the capture of multiple bands, which includes RGB sensors. While RGB images are traditionally referred to as *color images*, nowadays the term multispectral is more commonly found in literature as describing RGB. Even more so, the RGB+NIR modality, where an image contains four wide-band channels including near-infrared, is regularly described as multispectral. Therefore, it becomes increasingly common in literature to employ the term hyperspectral as soon as a significantly larger number of bands is involved. Traditionally, the term hyperspectral was used for sensors that either provide a very high spectral resolution, or image the electromagnetic spectrum outside the visible light, or both.

In this thesis, when we use the term multispectral, we refer to images with about 30 bands in the visible range, an example of which is depicted in Figure 2.2. For hyperspectral images, we assume hundreds of bands that may reach a typical range of 400 nm to 2400 nm.

2.1.1 Sensors

Multispectral or hyperspectral image data is often referred to as a cube. The first two axes of the cube refer to the image plane, while the z-axis denotes the spectral component. Several techniques exist to acquire such a cube, band-by-band, line-by-line, or pixel-by-pixel. Traditionally, acquisition takes a significantly longer time than RGB imaging. Recent advantages however lead us towards real-time capture, and therefore, video capture possibilities. Here we describe the three most prominent designs.

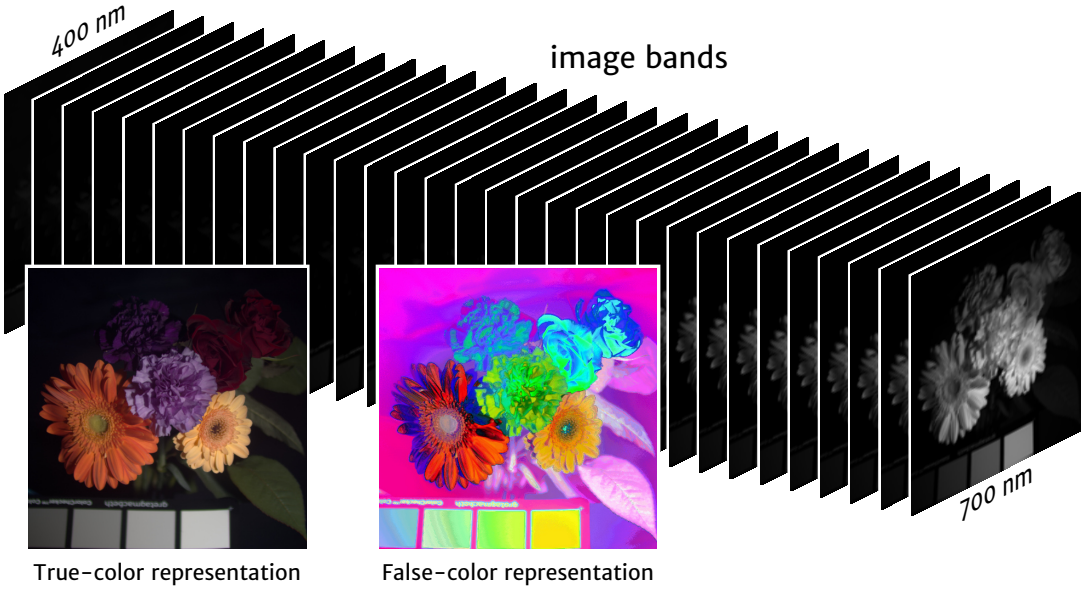


Figure 2.2: Illustration of an example multispectral image from the *CAVE* dataset. The image consists of 31 bands ranging from 400 nm to 700 nm. True-color and false-color representations are computed from the image data.

Line Scanners

A line scanner is typically referred to as push-broom scanner in aerial capture. It obtains the image cube line-by-line and is designed after a spectrometer. A prism between the lens and a CCD or CMOS chip splits the incoming light according to wavelength. On the sensor, each column then reflects the spectrum of a single point on the image plane. Mechanical movement of the sensor perpendicular to the line orientation allows the gradual capture of the image. In passive remote sensing, this is the satellite movement in the orbit. In industrial settings, typically a fixed sensor observes a moving production line. In the lab, a rotating mirror can be used [Herr 12]. Figure 2.3 shows the two components of such a hyperspectral sensor design by manufacturer SPECIM.

Push-broom scanners are typically hyperspectral, as a high spectral resolution can easily be obtained with a spectroscope. The drawback is that acquisition time grows linearly with vertical resolution. Images from push-broom sensors are typically obtained in the Band interleaved by line (BIL) format according to the image formation process.

Tunable Filters

Filter-based designs are typically found “on the ground”, but are also suitable for space and airborne imaging. Similar to the line scanner design, a monochromatic sensor chip is used. A filter is placed in front of the sensor that transmits a small wavelength band and excludes light of other wavelengths to obtain one image band. The image cube is obtained by subsequent captures with different filter settings. Early setups would use a filter wheel that is rotated between shots. Today, filter

Figure 2.3: Two components of a SPECIM hyperspectral sensor. The mirror unit to the left is attached in front of the lens seen on the right.

switching is done without mechanical work, using liquid crystal tunable filters (LCTF) or acousto-optical tunable filters (AOTF). They are electronically controlled and significantly improve capture speed and reduce misalignment between image bands.

A LCTF/AOTF-based sensor is more compact and less complicated to setup and operate when compared to push broom scanners. A major drawback is the low light efficiency of such filters. Long exposure times are needed to avoid underexposure, and acquisition time grows linearly with the spectral resolution. Especially in the case of AOTF, image quality is poor, for the benefit of faster tuning speeds and a broader wavelength range. Older reviews of such sensors are provided by Fisher et al. and Gat [Fish 98, Gat 00]. Lu and Fei, and Liang discuss more recent technology [Lu 14, Lian 12].

A recent design by Xerox PARC embeds an LCTF layer directly on the chip for a significant reduction in sensor size and increase in aperture. A drawback of their chip is that it produces wider bands especially in higher wavelengths [Hegy 15].

Single-Shot Designs

Other sensor designs that employ a custom chip in a traditional camera system recently became commercially available. They allow to capture the whole image cube in a single shot.

In the design by IMEA, spectral filtering is incorporated into the CMOS manufacturing process [Geel 14]. The bandpass filters are directly attached to the sensor and can therefore be applied on a per-pixel basis. As compared to other sensor designs, the manufacturer claims production costs similar to a regular, monochromatic CMOS chip. Several chip layouts are offered, including a mosaic layout. The chip is divided into 4×4 regions which captures one spectral band per pixel, obtaining a 16-band image. This is similar to the Bayer pattern of RGB cameras, where 2×2 regions on the chip capture the colors red, green, green, blue. One drawback of the current offering is a limited wavelength range.

Another recent design is based on transverse field detectors [Mart 14]. Here, the incoming light passes through several layers in the detector chip. In this case, spatial

	Wavelength (μm)	Characteristics and Uses
①	0.45–0.52	Bathymetric mapping (max. water penetration), distinguises soil from vegetation, deciduous from coniferous vegetation
②	0.52–0.60	Emphasizes peak vegetation, good for assessing plant vigor
③	0.63–0.69	Discriminates vegetation slopes
④	0.77–0.90	Emphasizes biomass content and shorelines
⑤	1.55–1.75	Discriminates moisture content of soil and vegetation
⑥	10.40–12.50	Thermal infrared: thermal mapping, estimated soil moisture
⑦	2.09–2.35	Hydrothermally altered rocks associated with mineral deposits

Table 2.1: Bands captured by LANDSAT Thematic Mapper and their characteristics [US G15, Gonz08, Chapter 1].

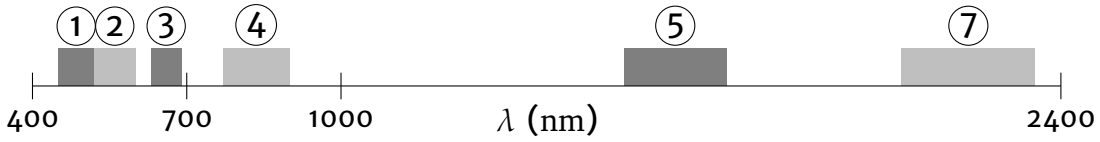


Figure 2.4: Bandpass wavelengths of LANDSAT Thematic Mapper for all bands in Table 2.1 except ⑥, which is far off-scale.

resolution is not sacrificed for spectral resolution, as the full spectrum is recovered in each pixel. In one capture shot, they capture up to 18 bands. However, the filter responses do not cover the spectrum as uniformly as LCTF.

These new designs are promising, as they are first to allow high quality video capture of a full image cube. They also further reduce production costs and design complexity.

2.1.2 Applications

Emerging sensor designs that reduce the cost to operate a multispectral or hyperspectral sensor and lift restrictions on their setup in non-lab environments are constantly widening the field of applications. Based on the hardware that most recently became commercially available, we can expect an even broader application scope of multispectral imaging in the near future. Here we give a non-exhaustive overview.

Remote Sensing and Astronomy

As mentioned earlier, multispectral and hyperspectral image capture has the longest tradition in remote sensing. An introduction to the topic is given by Shaw and Manolakis [Shaw02]. The NASA LANDSAT satellites are a prime example of high-quality earth imaging and were equipped with multispectral sensors early on. The LANDSAT 4, 5 satellites carry the *Thematic Mapper* instrument, capturing seven wide bands with a whisk-broom sensor (a precursor to the push-broom sensor with a rotating mirror scanning each line), while the still operational LANDSAT 7

captures an additional panchromatic band for increased spatial resolution [US G 15]. Table 2.1 lists the filter bands and their specific characteristics and uses. Figure 2.4 shows all filter bandpasses except thermal infrared on the wavelength scale. Since 2013, LANDSAT 8 is operating with further enhanced push-broom sensors and a total of 11 bands. The selection and characterization of bands for the Thematic Mapper give a good hint at the information of interest in remote sensing and the wavelengths relevant for it.

There is a plethora of applications in earth observation using image data from the LANDSAT thematic mapper, the Earth Observing-1 Mission (EO-1) satellite, the Advanced Spaceborne Thermal Emission Reflection Radiometer (ASTER) and other sources. High quality spectral libraries are publicly available containing thousands of spectra that allow for classification of minerals, rocks, terrestrial soils, manmade materials, meteorites, vegetation, snow and ice [Clar 07, Bald 09]. A recent, comprehensive review of applications and methods popular in hyperspectral remote sensing data analysis is provided by Bioucas-Dias et al. [Biou 13].

Instead of observing earth from space, multispectral data is also produced by observing space from earth. In astronomy, images are formed by combining the output of sensors that capture a wide range of electromagnetic emission features (i. e. intensities and wavelengths) of astrophysical objects through telescopes [Li 08]. Different properties come to light depending on the wavelength range. A well-known source of such data representations are the so coined allsky surveys that attempt to cover a major portion of the entire celestial sphere.

Medical diagnosis

A range of medical applications were studied for hyperspectral imaging, including, but not limited to, dermatological diagnosis, ophthalmology, detection of several cancer types, and diabetes. Lu and Fei give a broad review of applications and discuss clinical trials, in-vivo studies, and histological studies [Lu 14]. Sensors used for diagnosis typically measure reflectance, in some applications combined with fluorescence. In some cases, usually when integrated in microscopy, also the transmission of light is captured.

For diagnosing retinal diseases, a fundus camera can be integrated with a hyperspectral sensor, as presented by Johnson et al. [John 07]. A snapshot system captures 450 nm to 700 nm with 50 bands in about 3 ms. Through the hyperspectral data, functional maps are computed, e. g. to detect diabetes, venous occlusion, or shortage of blood flow via oxygen saturation. As related earlier, oxygen saturation of the blood is an indicator used to diagnose several medical conditions. Hyperspectral imaging of the human skin not only helps in detecting melanoma (malignant melanoma are a deadly form of skin cancer) or Kaposi's sarcoma. The observation of quantitative and qualitative changes in the level of skin oxygenation can also be used to diagnose patients in shock [Canc 06].

Detection in biochemical and morphological changes helps provide diagnostical information for different types of cancers. Major areas of research are measurement of the blood volume and oxygenation, in vivo examination of tissue surfaces, morphological and structural analysis of histological specimens and in vivo recognition of protein biomarkers on the cellular level. Numerous published studies discuss

recognition of cervical cancer, breast cancer, head and neck cancer, prostate cancer, and colon cancer [Lu 14].

Diabetic foot ulceration is a major complication of diabetes, where changes in large vessels and microcirculation of the foot are pivotal in its development. Greenman et al. use a hyperspectral sensor with high spectral resolution in the range of 400 nm to 500 nm to investigate the hemoglobin saturation in the forearm and foot for diagnosis [Gree 05].

Other domains where spectral characteristic of a tissue are valuable information are image-guided surgery or heart and circulatory pathology, e. g. the peripheral arterial disease or atherosclerosis [Lu 14].

Cultural Heritage

In cultural heritage, imaging spectroscopy goes back as far as the early 1990s. Most prominent in this area are the conservation and analysis of historical art and artifacts in archeology [Casi 99, Pico 07, Lian 12]. Cucci et al. present two case studies of Gentile da Fabriano's panel painting Polittico dell'Intercessione (c. 1425) in the church of San Niccolo, Florence and a tempera drawing on paper by Attilio Mussino made in 1911 [Cucc 11]. The acquired images deliver valuable information of the characteristics of pigmentation and dyes, helping in the goal of identifying pictorial materials and their distribution. A particular concern for study is how various parameters like the particle size, concentration and types of binding medium can influence pigment identification [Lian 12]. It is shown that surface dirt does not affect the shape of the spectrum. Under-drawings that may exist, retouched areas and the artist's technique can also be revealed. One resulting application can be artist identification and authentication. Color measurements are also obtained for monitoring the conservation of artworks, for example the cleaning of a contemporary mural [Marc 14]. Damages and past interventions can be detected, and the natural degradation in the course of time due to moisture, transportation or laser cleaning can be visualized by rendering an image under artificial illumination [Lian 12]. UV-fluorescence imaging is particularly helpful for recovering erased or faded writing, while a combination of multispectral imaging with other non-invasive imaging techniques such as laser scanning and photogrammetry or optical coherence tomography (OCT) can amplify analysis capabilities [Lian 12].

Multispectral imaging offers an important and effective non-invasive method for examination of historical documents [Kim 10, Hedj 13]. Research objectives are analysis of the chemical composition of the ink, recognition of latent patterns in a palimpsest (a manuscript page from which the text has been scraped or washed off for reuse), segmentation of the written text as well as detection of degradation signs. Degradation is one of the main problems rendering works difficult to decipher. Hedjam and Cheriet digitized over 110 multispectral images of writs between the 17th and 20th century [Hedj 13].

Food Quality and Safety

Gowen et al. review a large body of work on hyperspectral imaging in food quality and safety control [Gowe 07]. Hyperspectral sensors are employed in non-

Identifier	Amount, Type	Pixels	Bands	Range (nm)	Source
CAVE	32, lab	508 × 512	31	400 - 700	[Yasu 10]
Foster	8, nature	1340 × 1020	33	400 - 720	[Fost 06]
	1, lab	820 × 820	31	410 - 710	
Indian Pines	1, remote sens.	145 × 145	200	400 - 2500	[Purd 16]
D.C. Mall	1, remote sens.	307 × 1280	191	400 - 2475	[Purd 16]
Dürer	2, art scan	1087 × 908	557	400 - 900	[Stef 07]
		1040 × 850			
Harvard	50, urban	1392 × 1040	31	420 - 720	[Chak 11]

Table 2.2: Image datasets used in this thesis for illustration and empirical testing.

destructive food analysis, being faster and less expensive than previous approaches such as liquid chromatography or mass spectrometry. Contamination and defects are recognized, and overall consistency and quality classified. Freshness of a wide range of food products, including meat, common vegetables, and fruits, is monitored with classifiers on hyperspectral data in the visible to near-infrared (400 nm to 1000 nm) or infrared (1000 nm to 1700 nm) wavelength ranges. Nicolai et al. review studies on measuring fruit and vegetable quality, which includes factors like the bitter pit or bruises on apples, or firmness of peaches [Nico 07]. Elmasry et al. give an overview for objective meat quality evaluation. Hyperspectral imaging is used for quantifying and characterizing visual features of meat such as color, quality grade, marbling, maturity, and texture, but also for identifying the chemical structure and related physical properties of all types of meat [Elma 12]. Imaging spectroscopy is now increasingly gaining in traction in the food industry, as its major drawbacks, high hardware costs and relatively slow acquisition speed [Gowe 07], are mended by more recent sensors.

2.1.3 Image Datasets

As of today, multispectral and hyperspectral sensors are significantly more expensive and harder to operate than RGB cameras. The storage size of a spectral cube, often preserved with a high bitrate, is also a factor when sharing data. Therefore, less data is publicly available when compared to more common imaging modalities. Table 2.2 provides the characteristics of the datasets used in this thesis, which are further described below.

In remote sensing, data from the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) is very popular in benchmarking [Gree 98]. AVIRIS scans taken from ultra-high altitude reconnaissance aircraft were among the first publicly available hyperspectral remote sensing data. The *Indian Pines* image, captured in 1992 over the Purdue University Agronomy farm, is a prominent example, as it is well-understood, including ground-truth labels for classification [Baum 15, Tara 09]. Figure 2.5 depicts the ground-truth labels for *Indian Pines*. Per consensus in literature, pre-processing is performed on this image. Twenty bands are discarded as they

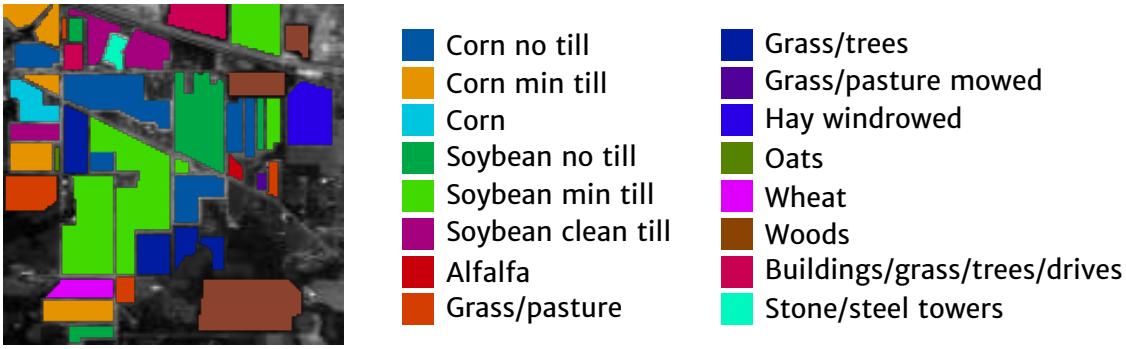


Figure 2.5: Ground-truth labels for *Indian Pines*. Labels overlay a grayscale representation of the image.

contain only atmospheric noise, resulting in only 200 of the 220 bands being used [Camp 05].

Another popular source of remote sensing data is the hyperspectral digital imagery collection experiment (HYDICE) sensor. The *D.C. Mall* image depicts the National Mall park in Washington D.C. and was captured in 1995. As with the AVIRIS image data, not all 210 bands are used due to some bands from near-infrared and infrared wavelengths are known to contained mostly noise due to water absorption in the atmosphere, and 191 bands remain. The image data can be obtained at [Purd 16]. Finally, often used for evaluating classification performance next to *Indian Pines* is the imagery of the Pavia university and city center taken by the Reflective Optics System Imaging Spectrometer (ROSIS) in 2002, with 115 narrow bands in the range 430 nm to 860 nm [Holz 03].

Several high-quality multispectral datasets are available that depict natural scenes. One was published in two parts, in 2002, and 2004, respectively by Foster et al. who studied the appearance of metamerism (i. e. materials of varied reflectance properties appear in same color) in natural scenes [Fost 06]. From these, we use outdoor scenes from the 2004 set and one indoor scene (the only lab scene) from the 2002 set. Recently, Foster et al. extended the set of available images and published new multispectral images of scenes undergoing natural illumination changes [Nasc 16, Fost 16]. Additionally, Chakrabarti et al. published an extensive dataset of indoor and outdoor scenes in daylight illumination for the goal of deriving statistics on multispectral images [Chak 11].

An example of hyperspectral imaging for art analysis and preservation is the scan of Albrecht Dürer's *Adoration of the Magi* by IFAC-CNR using a custom scanner [Stef 07, Pico 07]. It provides a high spectral resolution in the visible to near-infrared range as well as a very high spatial resolution of 289 ppi. The data is available in two forms: the full image with a spatial undersampling of 1:12 and a detail of the image in the captured resolution. Hedjam and Cheriet provide 21 multispectral images of historical documents written between the 17th and 20th centuries for the ICDAR 2015 MultiSpectral Text Extraction Contest [Hedj 15]. Eight bands in the ultra-violet, visible, and infrared wavelength range were captured with a filter-wheel camera [Hedj 13].

Lab images are also available from various groups. They have the advantage of providing considerably clean data which is helpful mostly for illustration, teaching and early testing of algorithms. The CAVE dataset by Yasuma et al. provides various objects in a lab setting [Yasu 10]. As the first four columns in each image consist of noise, these are trimmed for our use, hence the remaining resolution of 508×512 pixels. The Commonwealth Scientific and Industrial Research Organisation also provides lab images of good quality under a restrictive license [Data 16]. It dictates that the data is to be used only in conjunction with their Scyven software product.

Figure 2.6 depicts five example images that represent the different sensor types and image settings covered by the datasets used in this thesis. The images depicted are a *true color* representation of the respective original image data that relates human color perception. To obtain such a representation, the CIE 1931 colorimetric system and sRGB color space are used [Wysz 00, Chapter 3]. A detailed description of the computation will be given in Section 5.1.2 (see page 124). In this thesis we often show the true-color representation of an image along with algorithmic results on the image. It is an aid for understanding the depicted scene, yet the intuition given by such an image needs to be taken with a grain of salt: Obviously, many aspects of the original image data are lost in this representation.

2.2 Software for Hyperspectral Analysis

Alike multispectral and hyperspectral sensors, software for analysis of their output has a long history. One of the first widely recognized software packages, LARSYS, became available in the 1960s. It was operated via a text terminal. Several frameworks for graphical, interactive multispectral or hyperspectral data analysis that are still of broad influence today date back to the early 1990s. Earlier frameworks focus on a specific application, most prominently in the field of remote sensing. Boardman et al. provided an overview and the history of established software systems for remote sensing hyperspectral data [Boar 06]. Larry Biehl created an updated list on the web in 2007 [Bieh 07]. In this section we give first a brief overview of domain-specific software packages before reviewing more widely applicable software. We focus on the interactive inspection of images.

2.2.1 Remote Sensing

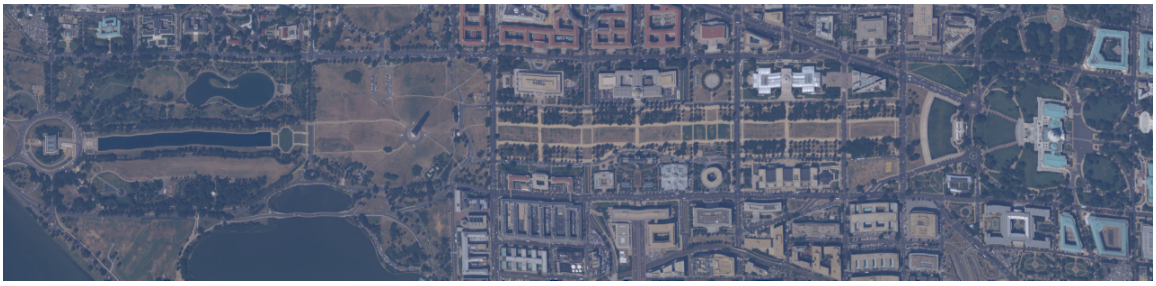
The Spectral Image Processing System (SIPS) was presented by Kruse et al. in 1992 [Krus 93]. It introduced the Spectral Angle Mapper (SAM), a tool still popular in the field of spectral matching [Denn 04]. SAM is used to compare observed spectra with a reference set for identification. Other included features were interactive contrast enhancement and spectral unmixing. Data was presented either as single bands or in a false-colored composite of three user-selectable bands. Additionally, the user could select a pixel to see its corresponding spectrum. Color-coded stacked spectra were provided for a selected slice (a vertical or horizontal line scan, or an arbitrary path in the image). This set of visualization forms is still typical in popular software for hyperspectral capture or analysis today.

(a) Fake and Real Peppers (CAVE)

(b) Dürer 1

(c) Cyflower (Foster)

(d) Harvard d3



(e) D.C. Mall

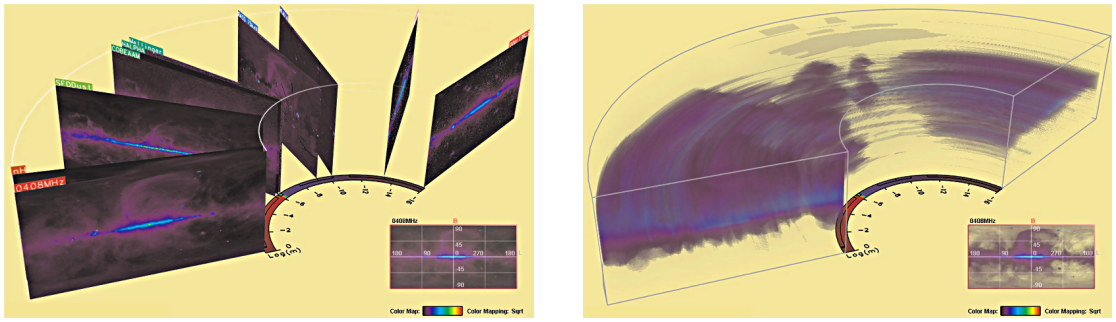
Figure 2.6: Example images from the multispectral and hyperspectral datasets used in this thesis. On (e), illuminant correction was applied for daylight illumination. Due to its high dynamic range, the contrast of (c) was enhanced for better visibility.

Being initially released in the same timeframe as SIPS, the MultiSpec freeware package by Landgrebe and Biehl is still under active development [Bieh 02]. MultiSpec can analyze multispectral images from various sources, using the versatile GDAL library for data I/O [Warm 08]. The focus of this software is classification. The target audience is the general Earth science community. MultiSpec provides common visualizations, e.g. single-band or false-colored spatial view, and single pixel spectral plots. Additionally, it can generate biplots that relate selected regions in a pair of bands and a statistics image display, which depicts the correlation between these bands.

A widely used commercial software is ENVI, initially developed by Boardman, Kruse and others [Exel 16, Boar 06]. The first version of ENVI was also released in the early 1990s. Several innovations in hyperspectral analysis were introduced in ENVI, including for example the Pixel Purity Index (PPI) [Boar 95]. PPI finds the most spectrally pure pixels in an image. These pixels typically correspond to endmembers (constituent spectra of an image) and are used in ENVI for endmember extraction. A notable innovation in visualization is the *n-Dimensional Visualizer*, which uses PPI as input. It is an interactive n-dimensional data visualization method that allows real-time rotation of scatterplots in n-dimensions. The presentation of n-dimensional scatterplots in 2-D can be somewhat unintuitive. Yet the tool is valuable to experts for identification of endmembers based on the depicted point clouds.

Two other notable pieces of software for hyperspectral remote sensing data analysis are HyperCube [Unit 16] and Opticks [Ball 16]. HyperCube is released by the U.S. Army Geospatial Center and contains functions to filter, warp (register two images), calibrate and undistort, photometrically project, and arithmetically manipulate the data. Opticks was originally developed by Ball Aerospace for the US Air Force and is freely available as open-source since 2007. It also includes many common tools like GIS annotations, false-coloring, and histograms. Furthermore, it incorporates the appealing concept of extensions, where third-parties can provide functionality through an external module. Thus, a range of available extensions adds further capabilities to the software. For example, the Spectral Processing Extension includes typical tools for hyperspectral data analysis [Ball 14]. Both HyperCube and Opticks are well-equipped, but do not provide alternatives to the common visualization paradigms present in SIPS, MultiSpec, or ENVI.

Within the remote sensing context, many works exist on visualizing a multispectral image by employing dimensionality reduction, which in itself is also often well-applicable to data from other sources. The goal is to present the image in false-color RGB, but not based on a simple combination of bands as in color composite images. Rather, dimensionality is reduced from spectral vectors to vectors of length 3, which are then used as `r`, `g`, `b` values. Based on the resulting pixel colors, the user should be able to discriminate regions of the image according to his specific interests. Methods to achieve this include variants of the Principal Component Analysis (PCA), Independent Component Analysis (ICA), as well as non-linear methods. Some variants also incorporate classification results, e.g. in thematic mapping. A recent approach by Cui et al. focuses on the interactive adaptation of such visualization [Cui 09]. We further discuss these methods in Section 5.1.3.



(a) image-stack model **(b)** volume-rendered model

Figure 2.7: Horseshoe model displays of astronomical data [Li 08]. A user-adjustable transfer function controls the opacity of data points in (b). Images courtesy of Hongwei Li.

2.2.2 Astronomy

Another field with a long history of hyperspectral imaging is astronomy. The ds9 software can be considered the most notable software framework in this field [Joye 03]. It is available from the Smithsonian Astrophysical Observatory and has its root in the SAOtng package, which dates back to 1995. This software is very powerful in the spatial representation of astronomical imagery. However, it provides a limited 3-D visualization of the data cube, where the z-axis may represent either the spectral domain, or the time domain in the case of consecutive monochromatic captures. The 3-D visualization does not utilize modern volume rendering techniques, making it rarely viable for display of non-astronomical data.

Li et al. explicitly tackled the question of how to present multi-band data [Li 08]. They drew image bands in 3-D space either as an image-stack or as a volume-rendered model, e.g. a horseshoe model. See Figure 2.7 for an example. Their volume rendering handles the obvious problem of clutter by applying transparency to individual data points based on their intensity or on a user-adjusted mask. This works especially well for astronomical data, which is rather sparse in the spatial domain. However, one cannot generally assume that large image regions may be faded out in such a manner for an unobstructed display of relevant data.

2.2.3 Other Application Domains

Multispectral imaging has become increasingly popular in the preservation and analysis of artwork as well as historical documents.

Colantoni et al. analyzed multispectral images of paintings from the perspective of human color perception [Cola 06]. From the image data, a representation in the CIE XYZ color space [Wysz 00, Chapter 3] is computed under controlled virtual illumination. Several tools can then be applied for visualization of trichromatic data. The original spectra are not considered in the analysis.

In 2010, Kim et al. presented a solution for interactive visualization of historical documents [Kim 10]. They provided a thoughtfully designed self-contained analysis tool and incorporated innovations in how the data is presented. Some are specific to

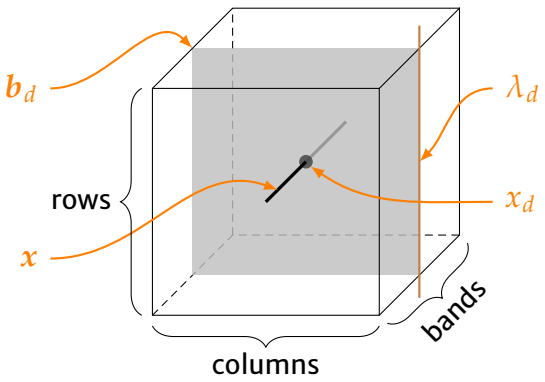
document analysis, e.g. dealing with aging-related artifacts, while others are more general. Within the view of a single spectral band, the user can select a region for which the data from a different band is displayed, referred to as the 'spectral lens'. Similarity maps can be computed between the mean spectral value of a selected region and all pixels from the image. The user can select a wavelength range over which the similarity is computed. However, similarity measures tailored to comparing spectra like SAM would be better suited for spectra comparison than the L_2 norm employed by the authors. The methods of Kim et al. are limited regarding the employed visualization techniques. Displays include single spectral bands, similarity maps, true-color images, and false-color images based on contrast-enhanced band fusion. A 3-D histogram plot is used to compare two spectra.

The Commonwealth Scientific and Industrial Research Organisation (CSIRO) offers the Scyven software free-of-charge which features the reflectance recovery and material classification pipeline developed at CSIRO [Habi 15, Robl 12]. Its visualization includes false coloring and an adaptation of the Parallel Coordinates visualization introduced in this work (see Section 5.3).

Labitzke et al. introduced an interactive framework for linear spectral unmixing of multispectral datasets [Labi 13a]. Spectral unmixing is especially popular in remote sensing, where the spatial resolution is often low. Many algorithms exist for finding endmembers and performing spectral unmixing [Kesh 02]. Labitzke et al. introduced an iterative method that can semi-automatically find endmembers. Then, visual feedback is provided by their complementary visualization that reflects the quality of the characterizing set. This set can be interactively modified in order to improve the unmixing. The authors explicitly differentiated their approach from our framework [Labi 13a, Jord 10]. Notably, the algorithm and workflow proposed by Labitzke et al. nicely complement the visualization capabilities discussed in Section 5.3.

2.3 Data Descriptors

The data behind a hyperspectral image, often termed "hyperspectral data cube", or shorter, "hypercube", consists of n_X pixels $\boldsymbol{x} \in \mathcal{X}$. Each pixel is a vector of spectral coefficients and has length n_D, whereas n_D is the number of bands captured in the image. Each coefficient x_d holds the sensor response in the corresponding band $\boldsymbol{b}_d$, with a center wavelength λ_d. The full width at half maximum (FWHM) is often also known for each band, but of limited concern for most applications.



In this section we describe how this data cube is formed and discuss alternative representations that may be computed from it.

2.3.1 Image Formation

The intensity values x_d that we obtain from a photosensitive sensor correspond to the incident irradiance on the area of a pixel on the sensor plane. In the notation of Angelopoulou, [Ange 00, Ange 07], for each point $\boldsymbol{p} = (x, y, z)^{\mathrm{T}}$ in the scene, the light incident on $\boldsymbol{p}$ is determined by its spectrum $\mathrm{e}(\lambda)$ and its direction relative to the point in the scene $\mathrm{E}(\boldsymbol{p})$. For this, we assume a single light source with a constant spectral distribution in all directions. Then, the amount of light reflected from point $\boldsymbol{p}$, denoted as $\mathrm{I}(\boldsymbol{p}, \lambda)$, results from a combination of the light incident on $\boldsymbol{p}$ and the surface reflectance $\mathrm{S}(\boldsymbol{p}, \lambda)$ of the surface at $\boldsymbol{p}$,

$$\mathrm{I}(\boldsymbol{p}, \lambda) = \mathrm{e}(\lambda) \cdot \mathrm{E}(\boldsymbol{p}) \cdot \mathrm{S}(\boldsymbol{p}, \lambda) \,, \tag{2.1}$$

where λ denotes wavelength. The reflectance function $\mathrm{S}(\boldsymbol{p}, \lambda)$ depends on the surface material, the scene geometry and the viewing and incident angles. Note that in the typical scene setup in hyperspectral capture, there is only a single light source present. This is direct sunlight in remote sensing satellite imagery or a controlled light source in most other applications. Models exist that deal with more light sources, e.g. a combination of direct and ambient illumination [Ries 09].

To obtain a consistent sensor response $x_d \approx \mathrm{I}(\boldsymbol{p}, \lambda_d)$ across the spectrum, sensors need to be calibrated. The calibration can be done with a spectral calibration light source and needs to be renewed frequently. However, in many analysis tasks, the goal of data capture is to obtain $x_d \approx \mathrm{R}(\boldsymbol{p}, \lambda) = \mathrm{E}(\boldsymbol{p})\,\mathrm{S}(\boldsymbol{p}, \lambda)$, i. e. the relative amount of light that would be reflected from an object into the sensor under flat illumination. In this case, data may be normalized against both wavelength-dependent sensor response variations and light spectrum at the same time using a Lambertian calibration target, e.g. the Spectralon [Geor 07]. The term Lambertian refers to purely diffuse materials [Hugh 13, Chapter 27].

In the example of the CAVE dataset (see Table 2.2), the coefficients x_d approximate $\mathrm{R}(\boldsymbol{p}, \lambda)$. To do so, a normalization was carried out by setting the exposure time for each band capture individually, such as to obtain the same response across bands on a target present in each scene. Due to limited sensitivity of the sensor at smaller wavelengths, however, the first three bands of these images remain underexposed, effectively introducing an artificial illumination that is flat across most bands.

2.3.2 Normalized Data

As described above, $\mathrm{R}(\boldsymbol{p}, \lambda)$ depends on scene geometry and the viewing and incident angles. For Lambertian surfaces, these effects do not interact with the surface material and changes of intensity induced by geometry are not depending on wavelength. A common method to diminish geometry effects therefore is the normalization of spectral vectors according to their magnitude in L_2,

$$\mathrm{N}(\boldsymbol{x}) = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_2} \,, \tag{2.2}$$

where $\|\cdot\|_2$ denotes the L_2 norm. The imperfection of this method lies in the reliance on Lambertian surfaces. With only very few exceptions (e.g. the

aforementioned Spectralon), materials in the real world do not exhibit purely diffuse reflectance.

2.3.3 Spectral Gradient

The spectral derivative was defined by Angelopoulou [Ange 00] as the partial derivative of the logarithmic image with respect to the wavelength λ,

$$\mathrm{L}_\lambda(\boldsymbol{p}, \lambda) = \frac{\partial(\ln(\mathrm{I}(\boldsymbol{p}, \lambda)))}{\partial \lambda} \,. \tag{2.3}$$

The spectral gradient is then the discrete approximation of spectral derivatives obtained by finite differences.

We may interpret the spectral gradient based on the Cook-Torrance reflectance model [Cook 82]. For purely diffuse surfaces, the spectral derivative is the normalized partial derivative of the surface albedo $\rho(\boldsymbol{p}, \lambda)$ offset by a constant illumination term c:

$$\mathrm{L}_\lambda(\boldsymbol{p}, \lambda) \approx \frac{\rho_\lambda(\boldsymbol{p}, \lambda)}{\rho(\boldsymbol{p}, \lambda)} + \frac{c}{\mathrm{e}(\lambda)} \quad , \tag{2.4}$$

where $\rho_\lambda(\boldsymbol{p}, \lambda) = \partial\rho(\boldsymbol{p}, \lambda)/\partial\lambda$ is the partial derivative of the albedo with respect to wavelength. Albedo itself is a material property, independent of illumination and geometry. In purely diffuse surfaces, the spectral gradient is therefore invariant to geometry. When the illuminant spectrum does not change over the topology of the image, this means that spectral gradient values stay constant within a material.

When not dealing with a purely diffuse surface, the Cook-Torrance model covers the dependence of specular reflectance on material properties in the Fresnel term. It turns out that, for purely specular reflectance, the spectral derivative becomes the normalized partial derivative of the Fresnel term offset by the same constant illumination term [Ange 07]:

$$\mathrm{L}_\lambda(\boldsymbol{p}, \lambda) \approx \frac{\mathrm{F}_\lambda(\boldsymbol{p}, \lambda)}{\mathrm{F}(\boldsymbol{p}, \lambda)} + \frac{c}{\mathrm{e}(\lambda)} \quad . \tag{2.5}$$

where $\mathrm{F}_\lambda(\boldsymbol{p}, \lambda) = \partial\mathrm{F}(\boldsymbol{p}, \lambda)/\partial\lambda$ is the partial derivative of the Fresnel term with respect to wavelength. The Fresnel term is also a material property, but unlike albedo it also depends on geometry [Hugh 13, Chapter 26]. Effectively, specular regions in the image can be separated from diffuse ones using the spectral gradient.

Often the spectral gradient is better suited for analysis of the captured scene when compared to the original spectral data. In the visual inspection of an image, the spectral gradient feature-vectors can offer a view that focuses more on data aspects that are directly related to material properties and the image formation process [Ange 07].

2.4 Similarity Measures

Many algorithms are based on measuring the similarity (or dissimilarity) of two data points. How this similarity is commonly measured is more varied in multispectral than in RGB.

2.4.1 Metrics

A *pseudometric* $d(\boldsymbol{x}, \boldsymbol{y})$ has the properties,

$$\begin{aligned} d(\boldsymbol{x}, \boldsymbol{y}) + d(\boldsymbol{y}, \boldsymbol{z}) &\geq d(\boldsymbol{x}, \boldsymbol{z}) , \\ d(\boldsymbol{x}, \boldsymbol{y}) &= d(\boldsymbol{y}, \boldsymbol{x}) , \\ d(\boldsymbol{x}, \boldsymbol{x}) &= 0 , \end{aligned} \tag{2.6}$$

whereas these conditions also imply that $d(\boldsymbol{x}, \boldsymbol{y})$ is non-negative. A *metric* or *distance function* $d(\boldsymbol{x}, \boldsymbol{y})$ is a pseudometric with the additional property,

$$d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y} . \tag{2.7}$$

In our context, it is desired for a dissimilarity measure to be a pseudometric, while the additional constraint of a metric is less relevant. Due to signal noise we do not expect to observe identical samples, rendering Eq. 2.7 ineffective.

A general approach to measuring dissimilarity of image pixels is to rely on a L_p-norm [Lee 07, Chapter 4], including the Manhattan distance L_1, the Euclidean distance L_2, and the less popular Chebyshev distance,

$$L_\infty(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_\infty ,$$

which intuitively is described as the maximum distance. In our work on supervised segmentation, we found that L_∞ is the best-suited of these three when comparing spectral vectors, with the additional benefit of a range independent of n_D. However, while the Euclidean distance is often used on spectra in literature (sometimes denoted as the Euclidean Maximum Distance [Kesh 04]), more specific measures were established for the comparison of two spectra $\boldsymbol{x}, \boldsymbol{y}$.

2.4.2 Spectral Matching

In spectral matching, prior knowledge exists through a library of material spectra. Observed spectra, e.g. single pixels, are then matched to their corresponding, or best fitting, library entry. Several similarity measures have been designed for this task, which implies that they should work well for discerning different materials in the scene. Robila and Gershman as well as Gutiérrez-Rodríguez et al. give a good overview [Robi 05b, Guti 10].

Back in 1992, the Spectral Image Processing System featured the Spectral Angle Mapper by Yuhas et al. [Yuha 92, Krus 93]. It employs the spectral angle between two spectra $\boldsymbol{x}$ and $\boldsymbol{y}$,

$$\mathrm{SA}(\boldsymbol{x}, \boldsymbol{y}) = \cos^{-1}\left(\frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{y}\|_2}\right) , \tag{2.8}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. SA effectively captures the angle between two spectra and therefore is invariant to pure intensity changes. It is a pseudometric: It is always greater or equal to zero; it is symmetric and follows the triangle inequality. However, it is not a metric due to $\mathrm{SA}(\boldsymbol{x}, a\boldsymbol{x}) = 0 , \forall a > 0$.

This property is shared by the Normalized Euclidean Distance, as proposed by Robila and Gershman [Robi 05b], given as

$$\mathrm{NED}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{d=1}^{n_D} \left(\frac{x_d}{\|\boldsymbol{x}\|_2} - \frac{y_d}{\|\boldsymbol{y}\|_2} \right)^2} \,. \tag{2.9}$$

Effectively, NED constitutes working in the normalized space as described in Eq. 2.2. In fact, SA and NED show quite similar behavior.

The Spectral Correlation Mapper (SCM) was defined by de Carvalho and Meneses in 2000, [De C 00], and is based on the sample Pearson correlation coefficient such as,

$$\mathrm{SCM}(\boldsymbol{x}, \boldsymbol{y}) = 1 - \frac{1}{2}\rho_{xy} = 1 - \frac{1}{2} \frac{\langle \boldsymbol{x} - \overline{\boldsymbol{x}}, \boldsymbol{y} - \overline{\boldsymbol{y}} \rangle}{\|\boldsymbol{x} - \overline{\boldsymbol{x}}\|_2 \cdot \|\boldsymbol{y} - \overline{\boldsymbol{y}}\|_2} \,, \tag{2.10}$$

where $\overline{\boldsymbol{x}}$ is the arithmetic mean of $\boldsymbol{x}$. SCM, as compared to SA, has the benefit of distinguishing between negative and positive correlations. Other versions of SCM are the Spectral Correlation Similarity, where negative correlations are cut off [Homa 04], or the Spectral Correlation Angle, formulated in a similar fashion to SA [Robi 05b].

Also in 2000, Chang introduced a measure that is based on information theory [Chan 00]. It is successfully applied in spectral matching scenarios [Robi 05b]. The spectral vectors are modeled as random variables. This interpretation covers the spectral variability of a pixel based on inter-band correlation. The random variable $\boldsymbol{p}^{(x)}$ of spectral vector $\boldsymbol{x}$ of length n_D is given as

$$\boldsymbol{p}^{(x)}_{1 \le d \le n_D} = \frac{x_d}{\sum_{e=1}^{n_D} x_e} \,. \tag{2.11}$$

We can then compute the Spectral Information Divergence (SID), which is based on the symmetric Kullback-Leibler information measure, as

$$\mathrm{SID}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{d=1}^{n_D} \boldsymbol{p}^{(x)}_d \log \frac{\boldsymbol{p}^{(x)}_d}{\boldsymbol{p}^{(y)}_d} + \sum_{d=1}^{n_D} \boldsymbol{p}^{(y)}_d \log \frac{\boldsymbol{p}^{(y)}_d}{\boldsymbol{p}^{(x)}_d} \,. \tag{2.12}$$

In 2004, Du et al. expanded SID by combining it with SA [Du 04]. As they are built on different interpretations of a spectral vector, the authors aim for an enhanced spectral discriminatory probability. Two versions of the combinations are proposed:

$$\mathrm{SIDSAM}_1(\boldsymbol{x}, \boldsymbol{y}) = \mathrm{SID}(\boldsymbol{x}, \boldsymbol{y}) \cdot \sin\left(\mathrm{SA}(\boldsymbol{x}, \boldsymbol{y})\right) \,, \tag{2.13}$$

$$\mathrm{SIDSAM}_2(\boldsymbol{x}, \boldsymbol{y}) = \mathrm{SID}(\boldsymbol{x}, \boldsymbol{y}) \cdot \tan\left(\mathrm{SA}(\boldsymbol{x}, \boldsymbol{y})\right) \,. \tag{2.14}$$

It was shown in literature as well as in our own experiments that the difference between SIDSAM_1, SIDSAM_2 is negligible. Therefore it is sufficient to concentrate on SIDSAM_1 from now on.

Robila and Gershman calculate the spectral angle on the spectral gradient, [Robi 05b], as

$$\mathrm{SGA}(\boldsymbol{x}, \boldsymbol{y}) = \mathrm{SA}(\boldsymbol{x}', \boldsymbol{y}') \,, \tag{2.15}$$

where $\boldsymbol{x}'$ and $\boldsymbol{y}'$ are the gradient vectors of spectra $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively. However, the benefit of combining the spectral gradient with the spectral angle measure is not well motivated.

In our work, we found that some graph-based image segmentation algorithms work best when the dissimilarities measured between spectral vectors are somewhat evenly spread out in their observed range, e.g. uniformly distributed. For various reasons, the effective distributions of dissimilarities obtained with some of the discussed measures can be more problematic to them than the low dynamic range, discretized L_2 distances obtained from trichromatic images, which these algorithms were designed for (we will visit this topic again in Section 4.2.2, Section 4.3.2). To improve results, we may employ histogram equalization [Gonz 08, Chapter 3]. It is a simple, yet effective technique to obtain a uniform distribution via transformation from any continuous probability density function (PDF). Given a vector of all measured dissimilarities $\boldsymbol{w}$, we obtain histogram-equalized weights $\boldsymbol{w}^*$ via

$$w_i^* = \int_0^{w_i} p_{\boldsymbol{w}}(\omega) d\omega \,, \tag{2.16}$$

where $p_{\boldsymbol{w}}$ is the PDF of all measurements and the right hand side of the equation is the respective cumulative distribution function. In our implementation, we approximate Eq. 2.16 via a cumulative histogram with 20 000 bins covering the range $[\min \boldsymbol{w}, \max \boldsymbol{w}]$.

2.5 Challenges

Our perspective on hyperspectral images stems from the computer vision discipline. In most computer vision algorithms operating on RGB data, color features play a minor role in relation to non-color features like shape or texture, as the information contained in a single `r`, `g`, `b` triplet is limited. For example, as noted in the beginning of this chapter, while skin detection can be performed reliably on multispectral data [Ange 01, Huyn 10a], it is an unsolved problem on RGB images when accounting for all the skin types under different illuminations, shadows, cluttered backgrounds and makeup [Kaku 07, Khan 12]. Especially cumbersome for classification algorithms are differences in illumination that cause the color measurement to be biased toward the color of the light source. This is why color constancy is an important topic in computer vision [Geve 12, Part III]. In hyperspectral analysis, we see paradigms that differ from color image analysis: Spatial context is of limited use, while on the other hand the information available per-pixel is rich and more reliable. Most hyperspectral methods in classification or object detection tasks work on single pixels as compared to larger image regions or even the whole image. Spatial context can be an assisting clue and features were proposed that incorporate the local pixel neighborhood. However we need to be aware that with multispectral sensors, we often see spatial resolution compromised for the sake of spectral resolution.

Furthermore, we may name two key characteristics of hyperspectral data: a high correlation between adjacent bands [Lee 93], and a comparably low signal-to-noise (SNR) ratio.

(a) 400 nm **(b)** 687 nm **(c)** 2202 nm

Figure 2.8: Three bands of *Indian Pines* image with enhanced contrast for better visibility.

(a) 420 nm **(b)** 560 nm **(c)** 720 nm

Figure 2.9: Three bands of cropped *Harvard d3* image with enhanced contrast for better visibility. The full image is depicted in true color in Figure 2.6d.

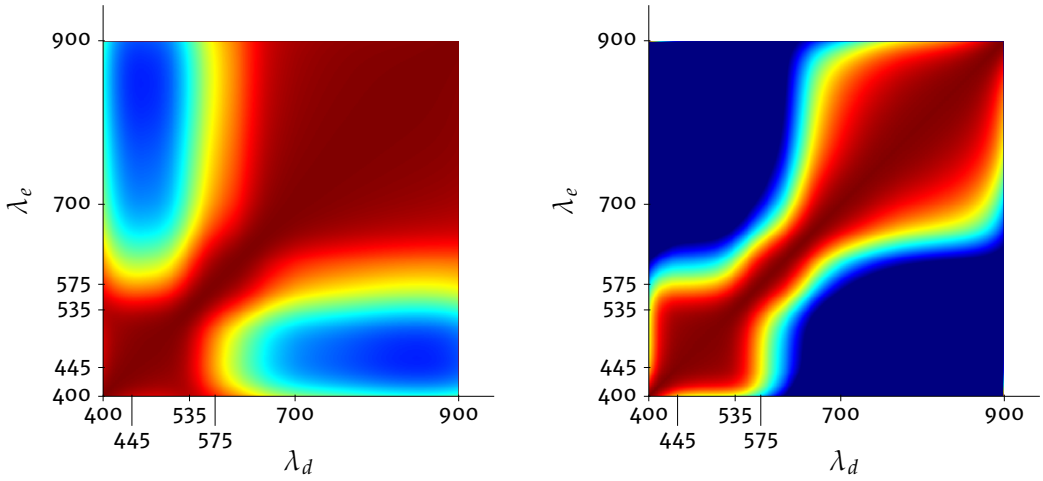
Hyperspectral images observe more noise when compared to monochromatic or RGB images as the higher spectral resolution reduces the amount of light that falls into the sensor per image band, necessitating longer integration times. An additional effect is a typically reduced sensitivity of sensors that capture the visible light spectrum up to near-infrared in the marginal parts of the captured spectrum. In general, the SNR is band-dependent. Figures 2.8 and 2.9 depict three bands of a hyperspectral remote-sensing image, and a multispectral indoor image, respectively. They show the variety of noise the respective bands contain.

The often high correlation between bands stems from a high spectral resolution paired with smooth reflectance curves. Whatever the redundancy, the high spectral resolution is needed in many specific applications where material reflectances of interest are accentuated only in a small part of the spectrum [Cucc 11].

To illustrate the inter-band correlation that is present in multispectral and hyperspectral images, [Tsag 05, Mano 08], we compute the sample Pearson correlation coefficient

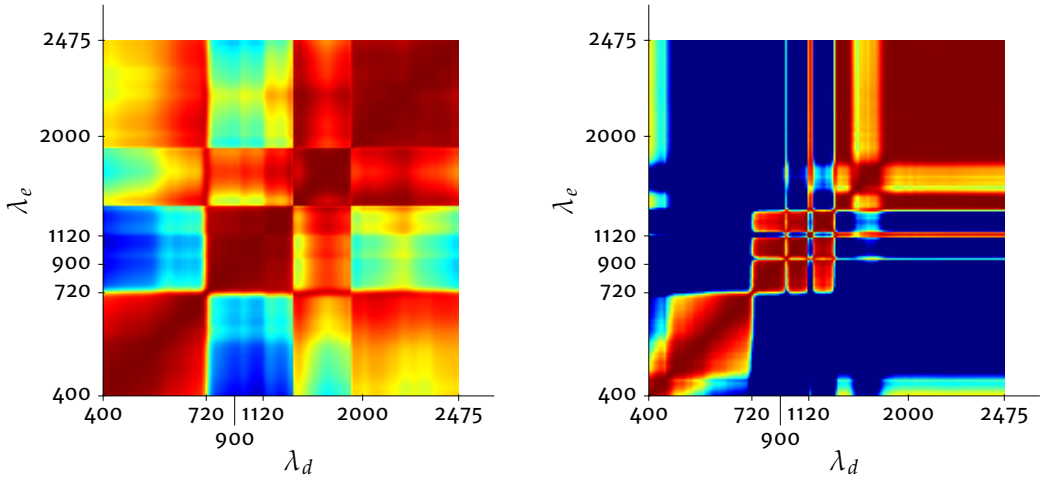
$$\rho_{de} = \frac{\mathrm{Cov}(\boldsymbol{b}_d, \boldsymbol{b}_e)}{\sqrt{\mathrm{Var}(\boldsymbol{b}_d)\,\mathrm{Var}(\boldsymbol{b}_e)}}\,, \tag{2.17}$$

where $\boldsymbol{b}_d$ is a vector containing the intensities of the dth image band ($x_d, \boldsymbol{x} \in \mathcal{X}$).



(a) Original image **(b)** Normalized image

Figure 2.10: Inter-band correlation coefficient maps of *Dürer 1* image.



(a) Original image **(b)** Normalized image

Figure 2.11: Inter-band correlation coefficient maps of *D.C. Mall* image.

Figure 2.10 shows correlation coefficient maps of the *Dürer 1* image. The labeled wavelengths of 445 nm, 535 nm and 575 nm are the empirically determined peak sensitivities of the three different sets of cones in the human eye [Gonz 08, Chapter 6]. The human eye is insensitive to wavelengths beyond 700 nm, which is the far end of the red light spectrum. In Figure 2.10a, we find a high inter-band correlation across a wide range of bands that cover the red to near-infrared range. Also high is the inter-band correlation in the blue to orange color range. Now compare this to Figure 2.10b, the correlation coefficient map of the normalized feature space as defined in Eq. 2.2. We conclude that a huge portion of the inter-band correlation is based on the overall brightness of the painted colors, and therefore average pixel intensity. After normalizing for the vector magnitude, the correlation coefficient map becomes more organized.

We observe a similar pattern with the *D.C. Mall* image, as seen in Figure 2.11b. In the normalized representation, Figure 2.11b, we see several wavelength ranges that share a high correlation, yet these groups of bands are clearly distinct from each other in what information they represent. These block structures are discussed in literature and can be exploited [Jia 99, Tsag 05]. The three major groups visible here are visible light, near-infrared and mid-infrared. The distinct groups of correlated bands in the infrared range stem from the effect of absorbers in the atmosphere, including H_2O and CO_2.

A typical approach to handle the combination of highly-correlated, yet noisy data is dimensionality reduction. The feature space is transformed in such a way as to significantly reduce its dimensionality by preserving the most relevant information while removing redundancy and noise. A second approach is to reduce the complexity either by establishing relationships of the data points in the image through the means of spatial layout, or by clustering in the high-dimensional feature space.

In the following chapters, we will exemplify: (a) how automated analysis can be assisted by a combination of these approaches when using a manifold learning method as a precursor to high-dimensional clustering; (b) how manual inspection can be assisted by an interactive visualization workflow that combines both original, high-dimensional data as well as culled or computed low-dimensional data.

Chapter 3

Dimensionality Reduction

Several factors make it hard to work with the image data in its original dimensionality. One is the empty space phenomenon, responsible for the so-called curse of dimensionality [Beye 99, Lee 07, Chapter 1]. The corners of a high-dimensional hypercube essentially become spikes. Given a sphere with equal center to the hypercube, and the sphere's diameter equaling the hypercube's side length, almost all the available volume in the hypercube accumulates on the hypercube's many spikes, outside the sphere. Also, in high dimensions, the discrimination power of a metric vanishes, as the distribution of norms in a given distribution of points tends to concentrate. Another factor is the computational burden. For example, the simple operation of computing the dissimilarity between two pixels observes a significant increase in memory access as well as machine instructions when comparing hyperspectral to RGB input.

To address these issues, dimensionality reduction is often applied in literature as a pre-processing step for feature extraction. In this chapter, we will first give a brief overview over both linear and non-linear techniques before going into more detail on Kohonen's self-organizing map (SOM), essentially a fast-to-compute discretized manifold learning method. We then discuss how we enhanced and generalized the SOM for our applications. We introduce a new probabilistic method for manifold learning, which is closely related to the SOM but formulated based on the Expectation-Maximization framework. To conclude, several aspects of these methods are discussed based on our experiments.

3.1 Related Work

The main distinction in dimensionality reduction methods is between linear and non-linear transformations. Considerably fast to compute and well-understood methods exist for linear feature space transformations that lead to the most significant information being concentrated in a low-dimensional flat. However, they are limited in preserving non-orthogonal relationships. The defining differences between two generally similar spectral responses in a scene are easily lost in such a transformation [Cher 03]. Non-linear methods operate with different constraints, or none at all, which makes them able to perform better in regards of preserving

and exposing dissimilarities between spectra. On the other hand, these are typically known to be computationally expensive to the point of becoming unfeasible for the number of data points present in an image. A good compromise is needed between speed and capability when choosing a method.

3.1.1 Linear Transformations

Linear dimensionality reduction is defined by a linear projection of the input data to a lower dimensional feature space. Principal Component Analysis (PCA) is the most prevalent of these algorithms and we will refer to it again in Chapter 5.

Principal Component Analysis

The PCA, also known as Karhunen-Loève-Transform, is a very well understood and broadly applied linear method in dimensionality reduction, lossy data compression, feature extraction, and data visualization [Bish 06, Chapter 12]. It is very prominent also in multispectral analysis as a feature extraction method for classification [Fauv 09], and for visualization [Jaco 05]. The key idea behind PCA is to find a rotation in the hyperspace $\mathbb{R}^{n_D}$ that captures the maximum spread of the data samples in the principal axes. Based on this rotation, data is projected onto a lower dimensional space $\mathbb{R}^{n_P}$, known as the principal subspace, by only preserving the first n_P vector coefficients. Bishop derives several PCA formulations [Bish 06, Chapter 12], here we illustrate the formulation based on maximizing the projected data variance.

Consider the case where $n_P = 1$. The direction of this one-dimensional space is defined by $\boldsymbol{u}_1$, which is a unit vector by choice. Each data point $\boldsymbol{x}$ is projected onto a scalar value $\boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{x}$. The goal is to maximize the variance of the projected data. The mean of the projected data is computed from the sample mean, $\boldsymbol{u}_1^{\mathrm{T}}\overline{\boldsymbol{x}}$. In the same fashion, the variance of the projected data is

$$\frac{1}{n_X}\sum_{i=1}^{n_X}\left(\boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{x}_i - \boldsymbol{u}_1^{\mathrm{T}}\overline{\boldsymbol{x}}\right)^2 = \boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{S}\boldsymbol{u}_1\,, \tag{3.1}$$

where $\boldsymbol{S}$ is the covariance matrix of the input sample set $\mathcal{X}$, given by

$$\boldsymbol{S} = \frac{1}{n_X}\sum_{i=1}^{n_X}(\boldsymbol{x}_i - \overline{\boldsymbol{x}})(\boldsymbol{x}_i - \overline{\boldsymbol{x}})^{\mathrm{T}}\,. \tag{3.2}$$

The projected variance $\boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{S}\boldsymbol{u}_1$ is now maximized with respect to $\boldsymbol{u}_1$ under the condition that $\boldsymbol{u}_1$ is a unit vector, i. e. $\boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{u}_1 = 1$. We obtain

$$\boldsymbol{u}_1' = \arg\max_{\boldsymbol{u}_1} \boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{S}\boldsymbol{u}_1 + \lambda_1(1 - \boldsymbol{u}_1^{\mathrm{T}}\boldsymbol{u}_1)\,, \tag{3.3}$$

where λ_1 is a Lagrange multiplier, which we use to reformulate the constrained maximization as an unconstrained one. The first derivative with respect to $\boldsymbol{u}_1$ yields a stationary point at

$$\boldsymbol{S}\boldsymbol{u}_1 = \lambda_1\boldsymbol{u}_1\,, \tag{3.4}$$

which implies that $\boldsymbol{u}_1$ is an eigenvector of $\boldsymbol{S}$. It also implies that the variance of the projected data is given by

$$\boldsymbol{u}_1^{\mathrm{T}} \boldsymbol{S} \boldsymbol{u}_1 = \lambda_1 \,. \tag{3.5}$$

To conclude, the solution to PCA with $n_P = 1$ is given by the eigenvector that corresponds to the largest eigenvalue of the original data's covariance matrix. This process can be repeated under the constraint that $\boldsymbol{u}_i \perp \boldsymbol{u}_j \,, \forall j < i$. In fact, the n_P projection vectors are the eigenvectors corresponding to the largest n_P eigenvalues of $\boldsymbol{S}$.

Other Methods

Two methods that are closely related to PCA and also used for dimensionality reduction on hyperspectral data are Linear Discriminant Analysis (LDA) [Band 09] and Projection Pursuit (PP) [Jime 99]. LDA uses class-labeled data samples to optimize class separability. The widely used Fisher criterion is based on maximizing the distance among the means of the classes while minimizing the intra-class variances. Projection Pursuit is based on a projection index, which is an objective function that can be based on class-labeled data [Jime 99]. However, also unsupervised variants of PP were applied on hyperspectral data. Iffaraguerri and Chang employ the information divergence between the distribution of projected data and the Gaussian distribution as a projection index [Ifar 00]. The rationale is that a bimodal distribution, i. e. a projection that separates clusters, yields a high divergence with normally distributed data. The PCA itself can also be cast into PP by setting the projection index as the projected data variance.

The Independent Component Analysis (ICA) is a well-understood algorithm for finding a linear mixture in the data that is also known as blind source separation [Lenn 01, Wang 06b]. In contrast to PCA, which leads to uncorrelated components, ICA leads to statistically independent components through a non-orthogonal projection. The independence criterion assumes that each component represents one of several statistically independent non-Gaussian source signals, which are linearly combined in the observed signal. ICA is related to PP, more explicitly to the method by Iffaraguerri and Chang, as it is able to compute projections which lead to the least gaussian-distributed projected data. The model of ICA is

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} \,, \tag{3.6}$$

where $\boldsymbol{x}$ is a vector of observed signals (e. g. a multispectral pixel), $\boldsymbol{A}$ is a matrix of mixing coefficients, and $\boldsymbol{s}$ is a vector of independent source signals. This formulation is particularly interesting from the perspective of spectral unmixing in hyperspectral analysis. In spectral unmixing, each pixel $\boldsymbol{x}$ is seen as a linear mixture of several material prototypes, or *endmembers*, stored as columns in $\boldsymbol{A}$. In this interpretation, $\boldsymbol{s}$ is then the vector of abundances in a model with no noise term [Wang 06b]. It should be noted that ICA is considerably slower to compute on hyperspectral data than PCA [Robi 05a] and by itself, ICA does not provide a strategy to select most relevant components for forming a low dimensional feature space [Cui 09]. Based on its premise, a low number of independent components only captures specific parameters of the high-dimensional distribution.

Finally, band selection is a popular method for reducing dimensionality in hyperspectral analysis. A selection of bands is preserved that maximizes the information for the task at hand while minimizing interference from unrelated sources. Prior knowledge is often employed to simplify the problem, e.g. physics-based band selection, or the use of labeled data or known spectral signatures [Yang 11, Guo 06]. Du and Yang propose a method for unsupervised band selection that follows a common scheme: Selection of an initial pair of bands, followed by iteratively adding more bands based on a dissimilarity measure [Du 08]. Keshava proposes an algorithm that uses the spectral angle between spectra for finding the initial band pair and adding successive bands [Kesh 04]. Other measures used to find most informative bands are variance calculated through PCA [Chan 99], entropy [Bajc 04], or mutual information [Wang 06a].

3.1.2 Non-linear Transformations

Several commonly used non-linear transformations are based on their linear counterparts, but formulated using the *kernel trick* [Bish 06, Chapter 6]. The original feature space is elevated to a higher-dimensional space using a transformation $\phi(\boldsymbol{x})$. Data that is not linearly separable in n_D becomes linearly separable through this mapping. Likewise, if the right transformation $\phi(\cdot)$ is found, it could express the relationship of two vectors $\boldsymbol{x}$, $\boldsymbol{y}$ within the manifold they are believed to lie on. This is done through a kernel function

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \left\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \right\rangle \ , \tag{3.7}$$

which is the inner product and therefore a symmetric function of its arguments in $\mathbb{R}^{n_D}$. The kernel trick now lies in the observation that for many methods, only the inner product of two variables $\boldsymbol{x}$, $\boldsymbol{y}$ is needed. We can express it through $\kappa(\cdot, \cdot)$, so no need arises to explicitly evaluate the mapping $\phi(\cdot)$.

A prominent kernel for Support Vector Machine (SVM) classification is the Gaussian kernel, [Bish 06, Chapter 6],

$$\kappa(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{y}\|_2}{2\sigma^2}\right) \ , \tag{3.8}$$

also known as the Radial Basis Function kernel, hinting at its radial symmetry [Camp 05]. It is designed to well-isolate Gaussian-distributed positive samples. It has been shown that the kernel trick can be used for almost all linear feature extraction methods, which includes Kernel PCA [Scho 02, Chapter 14], Kernel LDA [Scho 02, Chapter 15] and Kernel Projection Pursuit [Dund 04]. Unfortunately, it is typically not trivial to find the right kernel for a given problem without prior knowledge, so Kernel PCA and other kernelized methods may not yield good results for unseen data when using an off-the-shelf kernel. For example, when choosing the Gaussian kernel, its width σ^2 has to be tuned correctly to fit the data properly [Fauv 07, Chapter 1].

Other methods exist that capture nonlinear relations between different segments in the spectra. Lee provides a comprehensive overview of distance-preserving and

topology-preserving non-linear dimensionality reduction methodologies [Lee 07]. The optimal isometric mapping (ISOMAP) is an algorithm that was specifically used on hyperspectral data for visualization [Tene 00, Bach 06]. It seeks a manifold coordinate system that preserves geodesic distances in feature space. To do so, a neighborhood graph is constructed between samples either through Parzen windowing or k-nearest neighbors, using a pseudometric for edge-weighting (e. g. L_2 or SA). The distance between two samples $\boldsymbol{x}$ and $\boldsymbol{y}$ is then defined as the shortest path in the graph. Multidimensional Scaling (MDS) is then applied to find lower-dimensional coordinates, which works based on the pairwise distances of all samples. Approximations need to take place to make ISOMAP computationally feasible for larger images due to its $O\left({n_X}^2\right)$ memory and computation [Bach 06].

3.1.3 Self-organizing Map

The Self-organizing Map (SOM) was proposed in 1982 by Kohonen [Koho 82]. It is a single-layer neural network that converts the nonlinear statistical relationship between high-dimensional data into simpler geometric relationships [Koho 01].

In other words, a SOM provides a topological representation of the distribution of the underlying data, i. e. its manifold. It consists of a set of *model vectors* $\mathcal{M}$, which are arranged in a fixed low-dimensional topology. Model vectors are also often referred to as *neurons* and represent vectors in the original data space. Considering hyperspectral data samples $\boldsymbol{x} \in \mathbb{R}^{n_D}$, we obtain a model vector $\boldsymbol{m} \in \mathbb{R}^{n_D}$.

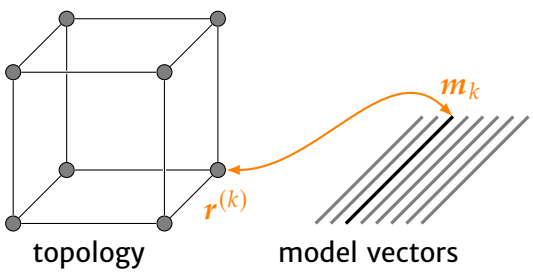
The SOM concept is based on the principle that for each input vector $\boldsymbol{x}$, we can find one distinct *best-matching unit* (BMU) $\boldsymbol{m}_c$ with index

$$c^{(x)} = \underset{k}{\operatorname{argmin}}\ \mathrm{d}(\boldsymbol{x}, \boldsymbol{m}_k)\,, \tag{3.9}$$

where $\mathrm{d}(\cdot,\cdot)$ is the Euclidean distance. In our experiments we did not observe any advantage by changing the distance function, e. g. to the spectral angle. The SOM is useful if for each input vector from the training data a BMU in close resemblance is found. Yet, model vectors should also be well-organized in a sense that they share a close relation with their local neighborhood in the SOM topology. This is important as the SOM topology is key to the dimensionality reduction.

Topology

Each model vector $\boldsymbol{m}_k$ comes with a *location* $\boldsymbol{r}^{(k)} \in \mathbb{Z}^{n_R}$, which has a one-to-one correspondence with k. Locations are a key ingredient to the training process and relevant in most use-cases of the SOM, as the relationship between model vectors is defined through location, essentially describing the layout of the learned manifold.



Typically, a 1-D integer lattice (2-connected) or 2-D grid is chosen for the model vector layout. In the 2-D case, a square lattice (4-connected) and a hexagonal lattice

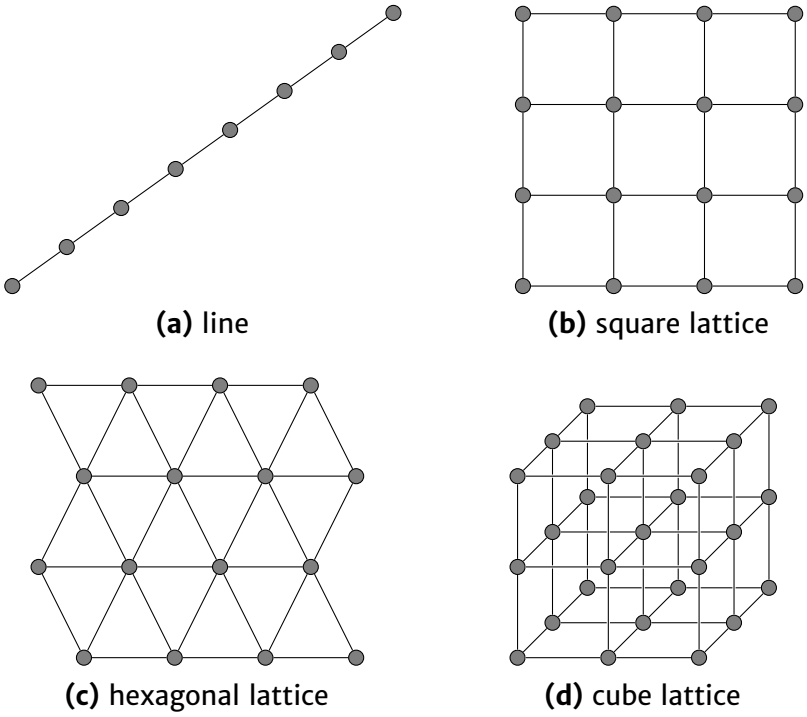
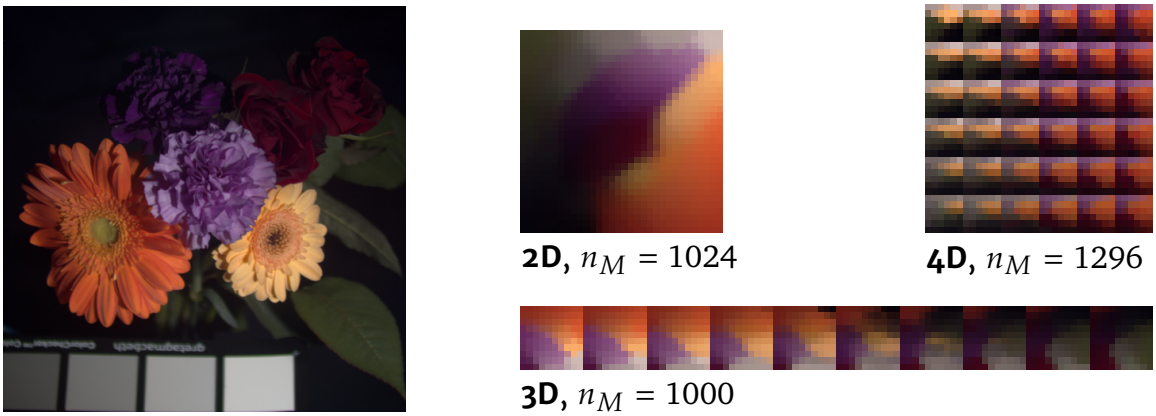


Figure 3.1: Common SOM topologies

(6-connected) are practical choices. The latter is useful when the SOM itself ought to be visualized. In our work we often rely on a less-common 3-D arrangement, which is a 6-connected 3-D lattice forming a cube [Jord 11, Jord 13a], and we also experimented with a tesseract (8-connected 4-D lattice). In Figure 3.1, four commonly used topologies are illustrated. In Figure 3.2, the training results of three SOMs on the same input data are visualized. For this visualization, all SOMs are mapped into a 2-D shape and colors are obtained by sRGB($\boldsymbol{m}_k$), which computes a true-color representation of a spectra (see Eq. 5.4 on page 124), the same method that is also used to visualize the input data.

A 1-D SOM provides a scalar index that may be used for global ordering [Toiv 03]. However, a 1-D lattice can be insufficient to adequately model the distribution found in a hyperspectral image, as exemplified later in Section 4.1.1. The third dimension on the other hand often enables the SOM to learn a more accurate topology on hyperspectral data when compared to a 2-D SOM.

Next to the underlying structure, the size of a SOM is an important choice. In its traditional use as a visualization tool, a SOM is typically rather small. Liu et al. evaluate SOMs of sizes 2×2 to 8×8 for feature extraction on time series [Liu 06]. Vesanto and Alhoniemi evaluate clustering of SOMs with sizes ranging from 16×13 to 24×19 [Vesa 00a]. Wijayasekara et al. use a $6 \times 6 \times 6$ cube topology for a 3-D visualization [Wija 11]. However, larger SOMs are used effectively in other domains [Skup 13]. Our applications also desire a larger SOM, which typically holds about $n_M = 1000$ model vectors. We will show in our experimental evaluation that we achieve fast training of a SOM of such a size. In general there is no reason to



(a) Input image **(b)** Self-organizing Maps

Figure 3.2: Visualization of SOMs after training on the *Flowers* image. Three SOMs were trained in the shapes of square, cube, and tesseract.

emphasize on one dimension over the other, so we employ a single side length in all dimensions. We denote it as

$$n'_M = \sqrt[n_R]{n_M}\,, \tag{3.10}$$

where n_R is the dimensionality of the chosen topology.

Training

The SOM training process is unsupervised and uses n_S unlabeled input samples. Typically, these come from the image to be processed. In certain scenarios, it is also worthwhile to train a SOM on a specific set of input spectra (which might be extracted from a collection of images) and then use it on several images.

At first, model vectors are initialized randomly,

$$\boldsymbol{m}_k \sim U_{n_D}(0,1)\,, \tag{3.11}$$

where $U_{n_D}(0,1)$ is the standard uniform distribution extended to $\mathbb{R}^{n_D}$. In our experience, variation of the initialization process is irrelevant, as model vectors are overwritten in the first few iterations.

Then, in each iteration $1 \leq s \leq n_S$, $s \in \mathbb{Z}$, we randomly draw a sample $\boldsymbol{x}$. We first determine the BMU $\boldsymbol{m}_c$ of input $\boldsymbol{x}$ given Eq. 3.9. Then, the BMU is updated by shifting it towards the new sample. Yet, the update should also affect model vectors in the SOM topology neighborhood of $\boldsymbol{m}_c$. This is the key to establishing the topological consistency in the SOM. We therefore update all model vectors as

$$\boldsymbol{m}_k^* = \boldsymbol{m}_k + \mathrm{h}_{c,k}(s) \cdot (\boldsymbol{x} - \boldsymbol{m}_k)\,, \tag{3.12}$$

where $\mathrm{h}_{c,k}(s)$ defines the influence of $\boldsymbol{x}$ on a model vector $\boldsymbol{m}_k$, given the current BMU $\boldsymbol{m}_c$. The influence is typically only dependent on a model vector's relationship with the BMU within the SOM topology, i.e. their coordinate distance. It is therefore commonly referred to as the *neighborhood function*. While, in theory, with each new sample we update all model vectors, the neighborhood function should confine the

effect of the update to a region around the BMU within the map. Kohonen [Koho 01] suggests to use the Gaussian kernel (Eq. 3.8),

$$\mathrm{h}_{c,k}(s) = \alpha(s) \cdot \exp\left(-\frac{\left\|\boldsymbol{r}^{(c)} - \boldsymbol{r}^{(k)}\right\|_2}{2\sigma^2(s)}\right), \tag{3.13}$$

where $\boldsymbol{r}^{(c)}, \boldsymbol{r}^{(k)}$ are locations of neurons c and k, respectively. The learning rate $\alpha(s)$ and kernel width $\sigma(s)$ are parameterizing the learning as explained below. Note that we use the Euclidean distance in the SOM topology, however, other distance functions may be used. One example would be a wrap-around distance simulating a torus to eliminate border effects [Kian 97].

Training Phases

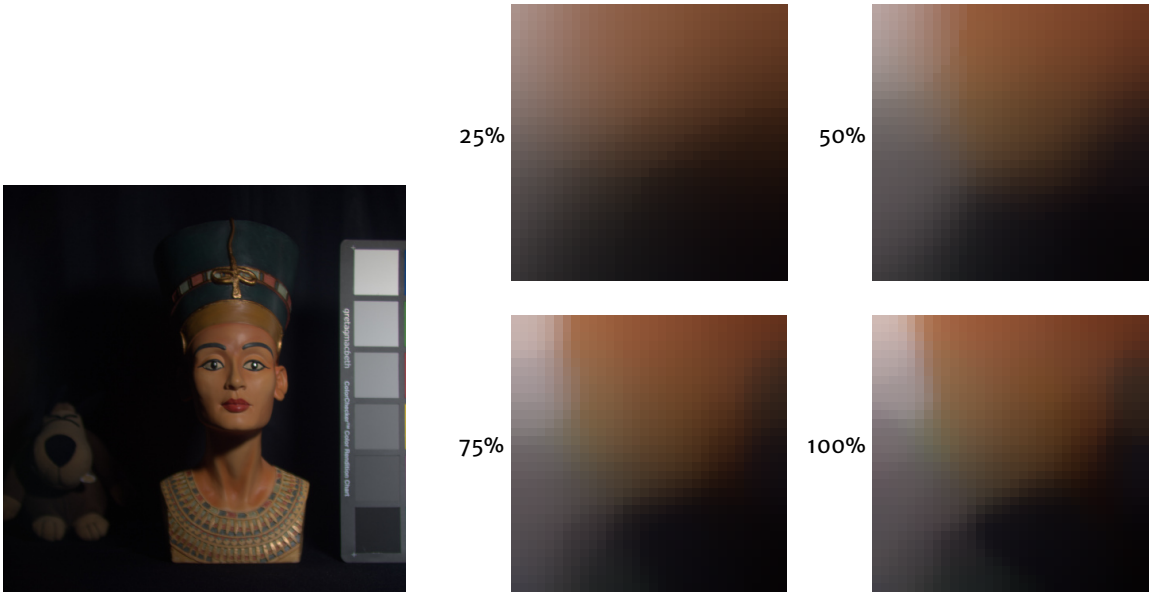
For the SOM training to be effective both globally and locally, the learning process is modified per-iteration through the learning-rate factor $\alpha(s)$ and kernel width $\sigma(s)$. The first describes the influence on all affected neurons, while the latter describes the sphere of influence in the SOM topology. Both functions $\alpha(s)$, $\sigma(s)$ are monotonically decreasing, in the same spirit of the "cooling" schedule known from metaheuristic optimization algorithms, e.g. simulated annealing or particle swarm optimization, where the temperature or the particle velocities are decreasing [Talb 09, Chapters 2, 3].

With this design, the training process can be seen as a combination of two phases with a seamless transition in-between. In the *orientation phase*, the global ordering of the SOM is determined. Model vectors find a general coverage of the manifold in feature space. Each new input sample has a broad influence, affecting the majority of SOM units. In the *refinement phase*, the influence of samples is reduced to local regions in the topology. This phase improves the individual representation of input samples and provides sharper separation of clusters in the data distribution. Eventually, $\alpha(n_S)$, $\sigma(n_S)$ lead to only the BMU being updated.

Figure 3.3 illustrates the training process of a SOM fed with samples from a multispectral image of the *CAVE* dataset [Yasu 10]. The SOM is trained with $n_M = 32 \times 32$ and snapshots of the SOM during training are depicted via sRGB($\boldsymbol{m}_k$). In the illustration it can be clearly seen that the SOM determines its global layout in the early stages of the orientation phase, while more subtle differentiations in the original distribution are best reflected after the later stages of the refinement phase. The initial map and first few training iterations are visualized in Appendix D on page 169.

3.2 Enhanced Self-Organizing Map

The SOM is very versatile in its primary application domain: a visualization tool for cluster analysis. It has a history as well of being employed for other uses, e.g. for classification, with adaptations carried out on the algorithm [Frit 94, Raub 02, Gopp 95, Kian 97, Lee 06]. This matches our experience that it can be a rewarding venture to extend applications of the SOM while carefully adapting it. In this



(a) Input image **(b)** SOM at four stages of training process (% samples)

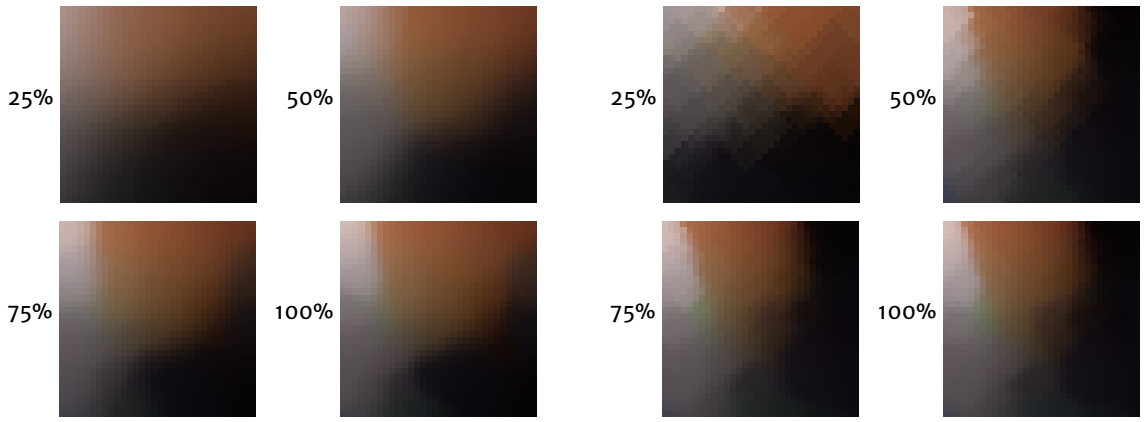
Figure 3.3: Visualization of SOM training progress on *Egyptian Statue* image. Both input image and SOM are rendered in true color.

section we introduce our own modifications to the SOM training, lookup, and implementation. Our modified algorithms are put to use in specific scenarios of hyperspectral image analysis and visualization [Jord 11, Jord 13a]. However, they are not bound to these applications.

3.2.1 Efficient Computation

In general, manifold learning is a computationally heavy task. However, quick learning is important for algorithms that learn the embedded manifold individually on the image data at hands. The SOM is a particularly efficient algorithm, as training takes $O(n_X n_M)$ time. The low complexity is due to using a discrete model, seen by the influence of n_M on the complexity class. The basic concept of finding a considerably small set of model vectors implies that the SOM provides a heavily quantized representation of the data. In most of our applications, we try to reduce this effect. A low number of neurons has especially adverse effects when operating with a higher-dimensional SOM topology, e.g. 3-D or 4-D. This is why, as compared to literature, we use considerably larger SOMs. This affects training times in two ways.

One, the number of distance computations in $\mathbb{R}^{n_D}$ to determine the BMU linearly increases with n_M. Software profiling reveals that distance calculation takes up the most significant share in processor time during training. Due to the high dimensionality, it is worthwhile to employ vector instructions, also known as Single Instruction, Multiple Data (SIMD) for distance computation. These instructions are applied on several data points simultaneously by the processor. When computing the distance $\|\boldsymbol{x} - \boldsymbol{y}\|_2$ using floating-point representations, we can group the coefficients of $\boldsymbol{x}$, $\boldsymbol{y}$ into packs of four for each, and execute instructions on a pair of packs at



(a) Gaussian kernel **(b)** uniform kernel

Figure 3.4: Visualization of SOM training progress for choices of $\mathrm{h}_{c,k}$ on the image depicted in Figure 3.3a. Maps are rendered in true color.

once as compared to a single coefficient pair. In early experiments, this gained a more than three-fold reduction in computation time for spectra with $n_D = 31$.

Two, with a larger SOM come increased kernel sizes, as an appropriate choice of $\sigma(0)$ is necessary to allow for a global organization of the map. As part of the update step, $\mathrm{h}_{c,k}(s)$ needs to be evaluated for every location $\boldsymbol{r}^{(k)}$ that is possibly affected. An early optimization is to start the update at $\boldsymbol{r}^{(c)}$ and then traverse the topology in a breadth-first fashion until the heuristic abortion criterion $\mathrm{h}_{c,k}(s) < 0.01$ is met. The calculation of $\mathrm{h}_{c,k}(s)$ includes evaluating the exponential function (see Eq. 3.13). It cannot be effectively pre-computed as the parameter $\sigma(s)$ is iteration-dependent. However we note that we use a radially symmetric kernel. We can exploit this symmetry in combination with the regular lattice arrangement of neurons. In the 2-D case, groups of four graph nodes each share the same distance to the BMU $\|\boldsymbol{r}^{(c)} - \boldsymbol{r}^{(k)}\|_2$, which implies they also share $\mathrm{h}_{c,k}(s)$. In the 3-D case, we obtain groups of eight graph nodes each that share this property. Therefore, we evaluate the neighborhood influence function once, then update four, or eight, nodes, respectively.

Finally, we note that choosing a uniform kernel can lead to an additional speedup. For this, we change the neighborhood influence to

$$\mathrm{h}_{c,k}(s) = \alpha(s) \cdot \left[\left\|\boldsymbol{r}^{(c)} - \boldsymbol{r}^{(k)}\right\|_1 \leq \sigma(s)\right] \,, \tag{3.14}$$

where $[\cdot]$ denotes the Iverson bracket. The Iverson bracket $[P]$ is 1 when the Boolean condition P is true, 0 otherwise [Knut 92]. This kernel is straightforward to optimize. The region of influence where $\mathrm{h}_{c,k}(s) < 0.01$ can be directly computed, and subsequently all nodes in that region can be updated at once. However, we need to test for the effect this change has on algorithmic performance. Figure 3.4 compares the behavior in training for both neighborhood functions on the *Egyptian Statue* image depicted in Figure 3.3a.

With the efficient training at hand, we can differ considerably from literature in our choice of n_M and realize the SOM shapes that our algorithms demand. Training times will be experimentally investigated in Section 3.4.1.

3.2.2 Ranked BMU Lookup

In most applications, the SOM is a visualization tool. Attributes of the model vectors themselves are presented, e.g. the unified distance matrix that contains the distances from each model vector to all of its neighbors [Koua 03]. It helps to visually identify clusters in the data.

When analyzing the data with the SOM, each input sample is assigned to a BMU given Eq. 3.9. In training this is an important property to ensure that the SOM finds an order. In the lookup phase, however, it reveals the major weakness of the SOM, which is its strong quantization of the input space based on the relative small number of model vectors n_M. The quantization puts a constraint on the quality of algorithms that use the SOM as input for distance calculation and dimensionality reduction [Gopp 95]. For example, a moderately sized 3-D map of $n_M = 64$ can only distinguish $n'_M = 4$ discrete values in each output dimension [Gorr 12]. In our own work, we train a considerably faster 3-D SOM with $n_M = 1000$, yet it only provides $n'_M = 10$ discrete values per dimension.

We reduce the effect of quantized output by employing a different look-up method [Jord 13a]. The idea is to incorporate more of the information available in the SOM, rather than from only one neuron. It goes hand-in-hand with the employment of a larger map size than typically used.

Instead of using a single best-matching unit, we develop a set of best-matching units (BMUs) as in [Sjob 09]. Furthermore, we order the BMUs according to the L_2 distance and assign a set of pre-determined weights to the ordered set. We coin this method as *ranked BMU lookup*. Rank-based weights are crucial. While a simple unweighted combination would only smooth the result, it is not reliable to use the L_2 distances directly as weights [Beye 99]. In the high-dimensional space, distances would appear very close to each other and the weights would not discern well. We define rank weights instead, to ensure both a majority contribution by the first BMU and significant contributions by the additional BMUs.

Consider a vector of BMU indices

$$\boldsymbol{c}^{(\boldsymbol{x})} = \underset{\boldsymbol{k}}{\operatorname{argmin}} \sum_{j=1}^{n_C} d(\boldsymbol{x}, \boldsymbol{m}_{k_j}) \,, \tag{3.15}$$

where n_C is the number of desired BMUs. For each pixel $\boldsymbol{x}$, we calculate a *representative location* $\boldsymbol{r}'$ as

$$\boldsymbol{r}' = \sum_{j=1}^{n_C} w_j \cdot \boldsymbol{r}^{\left(c_j^{(\boldsymbol{x})}\right)} \,, \tag{3.16}$$

given weights w_j. This provides a weighted average over locations of the n_C BMUs. Note that while $\boldsymbol{r}^{(k)}$ is discrete, $\boldsymbol{r}'$ is not. It describes the position in the learned topology that best represents $\boldsymbol{x}$. We propose two definitions for the weight vector $\boldsymbol{w}$ based on n_C and the condition

$$\forall \boldsymbol{m}_{c_j}, j < n_C : d(\boldsymbol{x}, \boldsymbol{m}_{c_j}) \leq d(\boldsymbol{x}, \boldsymbol{m}_{c_{j+1}}) \,, \tag{3.17}$$

which expresses that the BMUs are sorted according to distance to the query vector.

Gaussian Curve

An intuitive source for rank weights is the Gaussian curve,

$$w_j = \frac{1}{\|\boldsymbol{w}\|_1} \exp\left(-\frac{j^2}{2\sigma^2}\right) , \tag{3.18}$$

whereas $\sigma = \frac{n_C - 1}{3}$ is chosen such that the n_C weights contain 95% of the area under the curve.

Geometric Progression

A less obvious choice stems from exponential ranking selection in Genetic and Evolutionary Algorithms, which was also employed expediently in Particle Swarm Optimization [Blic 95, Jord 08]. It is the geometric progression,

$$\begin{aligned} w_j &= 2w_{j+1} , \\ \sum_{j=1}^{n_C} w_j &= 1 , \end{aligned} \tag{3.19}$$

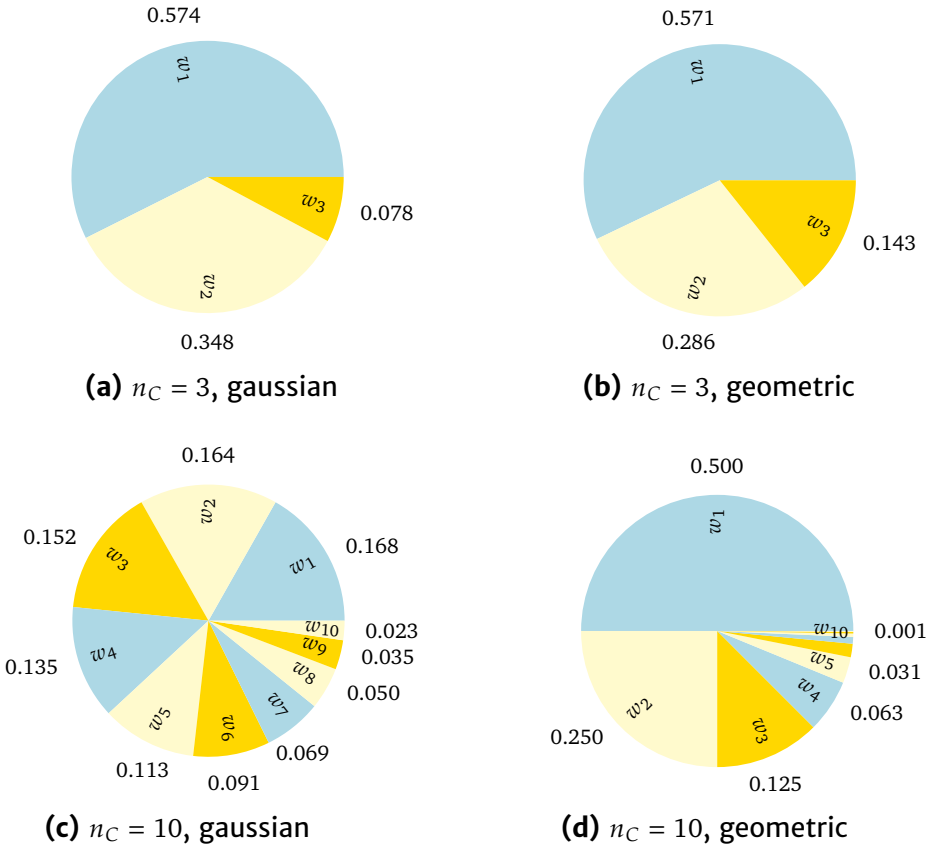
which states that the weight for rank j is always twice as high as the weight for subsequent rank $j+1$. It has two useful properties regarding the choice of n_C: One, $w_1 > 0.5$ for all choices of n_C. Two, weights shrink exponentially. Effectively, any choice of $n_C > 10$ will result in only insignificant variation in $\boldsymbol{w}$. This makes the rank weighting effectively parameterless.

Figure 3.5 shows the rank weights resulting from the two strategies side-by-side. In the case of the Gaussian curve, weights fall flat for large n_C, while in the case of the geometric progression, they stay sharp and the influence of low-rank BMUs diminishes.

Note that the lookup of a set of best-matching units as compared to a single BMU in a traditional SOM poses no significant additional burden in computation. It can be efficiently implemented with a heap data structure [Tama 99], and the number of (expensive) distance computations stays the same. As compared to a direct weighting scheme, rank weights have the advantage of being pre-computed.

3.2.3 Semi-supervised Training

Due to the stochastic nature of the training process, subsequent training runs will result in a significantly changed organization between the trained SOMs, even if trained on the same data. This can be attributed to varying initializations of the random number generator selecting input samples. However, this behavior does not necessarily imply that the learned structure is arbitrary. Instead, the majority of differences between subsequent training runs can be described as an improper rotation in the topology space. In some scenarios, it is desired to obtain a consistent orientation of the topology such as that certain spectra can be expected at the same position in several SOMs.



(a) $n_C = 3$, gaussian **(b)** $n_C = 3$, geometric

(c) $n_C = 10$, gaussian **(d)** $n_C = 10$, geometric

Figure 3.5: Comparison of rank-weighting strategies for three and ten representatives, respectively.

Another issue is the representation of data in the SOM. Due to the random selection of samples from the input, the representation of each data point depends on its relative frequency in the training data. It might be desired to guarantee the representation of certain spectra that occur relatively rarely, but are important for a given application.

We propose a semi-supervised training algorithm that can take into account known, labeled spectra. The training is then divided into two stages, resembling the two training phases outlined in Section 3.1.3: First, in the orientation phase, labeled spectra are fed. Second, in the refinement phase, unlabeled spectra are fed.

Supervised Orientation Phase

In the supervised orientation phase, for a given set of n_L labels $\mathcal{L}$, we seek to associate a certain position in the SOM topology to each label. Spectra of such a label should then concentrate around this position associated with the label. When designating a location for each label or class, it should not hinder the SOM's ability to properly project relations in the high-dimensional feature space onto the fixed topology. For example, in a linear SOM, a fixed anchor for class A at position 0, B at position 0.5 and C at position 1 would only be proper if the cluster of B lies between the clusters of A and C. However, consider a square lattice and classes A, B, C at the corners $(1, 0)$, $(0, 0)$, and $(0, 1)$, respectively. In this scenario, the SOM

neurons in the lattice area opposite to B (around $(1,1)$) are able to reproduce a transition between the clusters of A and C. It turns out that such a construction is always possible if the dimensionality of the SOM is chosen to be at least $\log n_L$. We then designate a corner of the map for each label. In the case of up to eight labels, we train a cubic SOM, whereas if up to 16 labels are known, we train a SOM of tesseract shape.

A naive approach to influencing the SOM organization in such a way could be to alter the initialization (see Eq. 3.11). However, due to the strong learning coefficients and large radii of influence in the first training steps, the effect of initialization is diminished. Instead, we control the orientation phase. We introduce a *magnetism* into the BMU determination,

$$c^{(x)} = \underset{k}{\operatorname{argmin}}\ \omega_l(\boldsymbol{r}^{(k)})\,\mathrm{d}(\boldsymbol{x}, \boldsymbol{m}_k)\ , \tag{3.20}$$

where l is the known label of $\boldsymbol{x}$ and $\omega_l(\boldsymbol{r})$ is a weight function based on the distance of $\boldsymbol{r}$ to the corner assigned to l, e. g.

$$\omega_l(\boldsymbol{r}) = \frac{4}{\sqrt{n_R}} \exp\left((\boldsymbol{r}_l - \boldsymbol{r})^{\mathrm{T}}(\boldsymbol{r}_l - \boldsymbol{r})\right)\ , \tag{3.21}$$

where $\boldsymbol{r}_l$ is a binary vector given by the numeral conversion of l to base 2. This expresses a strong incentive for model vectors close to a label-assigned corner of the lattice to represent pixels of that label. A second form we investigate is

$$\omega_l(\boldsymbol{r}) = \begin{cases} 1 & \|\boldsymbol{r}_l - \boldsymbol{r}\|_\infty < 0.5 \\ 1000 & \text{otherwise} \end{cases}\ , \tag{3.22}$$

which expresses a strong incentive for model vectors in a label-assigned orthant (e. g. quadrant, octant) of the lattice to represent pixels of that label.

Unsupervised Refinement Phase

In the refinement phase we proceed alike the traditional SOM. Random samples from the input data are fed. These samples are unlabeled. They enable the SOM to further establish relations within the input distribution and learn a manifold that spans naturally over all observed classes.

We obtain a map with two powerful properties: One, the orientation and layout of the topology is deterministic for all classes that were defined in the orientation phase. The position of a BMU for these classes therefore becomes canonical even in multiple trained SOMs. Two, a balanced representation is achieved for all said classes. A class that is under-represented in the training input is boosted in representation through the supervised orientation phase and may gain considerably in recognizability.

3.3 Probabilistic Manifold Learning

A common critique of SOM-based algorithms for classification or manifold learning is that no general proof exists for convergence [Cott 16], or that the SOM training

indeed leads to a topology that resembles the original distribution well enough. Rather, the fitness of the SOM for these applications is determined empirically.

We explore an alternative view on the manifold learning task that closely resembles some core concepts of a SOM, yet is based on a well-understood probabilistic model. Using a set of parameter vectors, we describe a graph whose vertices correspond to a mapping of the image spectra onto a feature space. The parameters are produced by a generative model, inferred from the image data using the Expectation Maximization (EM) algorithm [Bish 06, Chapter 9]. The theory in this section was developed jointly with Antonio Robles-Kelly [Jord 14].

In our model, we have three components. One, the observable spectra. We express them through the graph $G_M = (\mathcal{X}, \mathcal{E}_M)$ with nodes $\mathcal{X}$ comprised of input samples $\boldsymbol{x}_i \in \mathbb{R}^{n_D}$. The edges $\mathcal{E}_M$ then represent the connectivity between spectra. Theoretically, $\mathcal{E}_M$ is given by a local neighborhood relation on the manifold supporting the input spectra. The previously discussed ISOMAP algorithm attempts to determine $\mathcal{E}_M$ directly, but is slow to compute as finding neighborhood relations is a hard problem.

Two, a second graph $G_Q = (\mathcal{R}, \mathcal{E}_Q)$ that is linked to G_M. In contrast to G_M, it is constructed and has a fixed topology that is independent of the image data. We will establish a random field on this graph that follows the Gibbs distribution. The joint probability in such a Gibbs Field can be expressed by the product of clique potentials [Bish 06, Chapter 8]. A clique is any induced subgraph of a graph that is complete.

Three, a set of parameters Ξ, with $\boldsymbol{\xi}_k \in \mathbb{R}^{n_D}$. These parameters need to be estimated. The two graphs G_M, G_Q are realized in disjoint manifolds, $M \in \mathbb{R}^{n_D}$ and $Q \in \mathbb{R}^{n_R}$, respectively. The metrics on the two manifolds can be used to define the affinities between vertices within each graph. Parameters Ξ lie on M, but are connected to G_Q. Through them, G_Q will help us derive the topology of the manifold M.

We start by defining a link between the vertex-sets for G_M and G_Q,

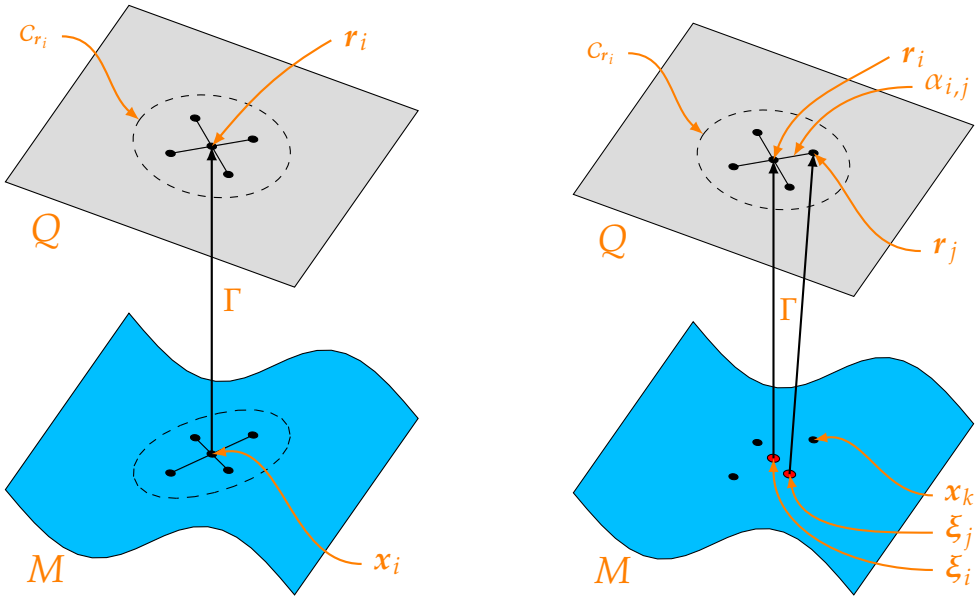
$$\boldsymbol{r}_j = \sum_{\boldsymbol{x}_k \in \mathcal{X}} [\boldsymbol{r}_j \sim \boldsymbol{x}_k] \Gamma(\boldsymbol{x}_k) \,, \tag{3.23}$$

where $\Gamma(\cdot)$ is a mapping function such that $\Gamma : \mathbb{R}^{n_D} \rightarrow \mathbb{R}^{n_R}$ and $[\cdot]$ denotes the Iverson bracket, i. e. $[\boldsymbol{r}_j \sim \boldsymbol{x}_k]$ is unity if $\boldsymbol{r}_j$ is adjacent to $\boldsymbol{x}_k$ and zero otherwise. This adjacency is not directly known. See Figure 3.6a for an illustration of this concept.

We now view the vertices of G_Q as random variables and G_Q as a Gibbs field. Consider the set of cliques $\mathcal{C}_{\boldsymbol{r}_i}$ containing vertex $\boldsymbol{r}_i$ (depicted in Figure 3.6a, where G_Q is a 4-connected lattice). The conditional probability of $\boldsymbol{r}_i$ can be written as a (weighted) product of pairwise potentials,

$$P(\boldsymbol{r}_i | \mathcal{C}_{\boldsymbol{r}_i}) = \frac{1}{Z_Q} \prod_{\boldsymbol{r}_j \in \mathcal{C}_{\boldsymbol{r}_i}} f_Q(\boldsymbol{r}_j, \boldsymbol{r}_i) \,, \tag{3.24}$$

where Z_Q is the partition function and $f_Q(\boldsymbol{r}_j, \boldsymbol{r}_i)$ is the potential function between the vertices $\boldsymbol{r}_j$ and $\boldsymbol{r}_i$. The potential function $f_Q(\boldsymbol{r}_j, \boldsymbol{r}_i)$ can be effectively viewed



(a) graph relationship **(b)** parameter vectors

Figure 3.6: Illustration of dual-graph concept. The relationship between nodes of the distinct graphs covering the two manifolds in (a) is relaxed using the parameter vectors in (b).

as the edge weight between $\boldsymbol{r}_j$ and $\boldsymbol{r}_i$. This will later provide a means to perform a maximum-likelihood estimation (MLE) on the vertex-set $\mathcal{X}$ in G_M based on the vertices $\mathcal{R}$ in G_Q. This process can be constrained using the metrics on manifolds M and Q. In Eq. 3.23, each vertex $\boldsymbol{r}_j$ is expressed as a sum over the product $[\boldsymbol{r}_j \sim \boldsymbol{x}_k]\Gamma(\boldsymbol{x}_k)$. For an unknown correspondence $\boldsymbol{r}_j \sim \boldsymbol{x}_k$, Eq. 3.23 can be relaxed using the parameter set Ξ which has a bijection to the vertex set $\mathcal{R}$. Effectively, $\boldsymbol{\xi}_j$ represents the j^{th} component of a mixture. With the prior η_j we obtain

$$\boldsymbol{r}_j = \sum_{\boldsymbol{x}_k \in \mathcal{X}} \eta_j \kappa(\boldsymbol{x}_k, \boldsymbol{\xi}_j)\,, \tag{3.25}$$

where $\kappa(\cdot)$ is a kernel function. This relaxation process is illustrated in Figure 3.6; in Figure 3.6b, the vertices in G_Q correspond to parameter vectors $\boldsymbol{\xi}_j$ which are supported by the vertex set $\mathcal{X}$. By substituting Eq. 3.25 into Eq. 3.24, we obtain

$$P(\boldsymbol{r}_i | C_{\boldsymbol{r}_i}, \mathcal{X}) = \frac{1}{Z_Q} \prod_{\boldsymbol{r}_j \in C_{\boldsymbol{r}_i}} f_Q \left(\sum_{\boldsymbol{x}_k \in \mathcal{X}} \eta_j \kappa(\boldsymbol{x}_k, \boldsymbol{\xi}_j), \boldsymbol{r}_i \right)\,, \tag{3.26}$$

which implies that by using the cliques in G_Q along with vertex sets $\mathcal{X}$ and $\mathcal{R}$, we can obtain a MLE of parameters Ξ based on our choice of kernel and potential function.

3.3.1 Parameter Learning

The potential function $f_Q(\cdot)$ is defined in the probability space corresponding to $\mathcal{R}$, i. e. the graph G_Q, whereas the kernel $\kappa(\cdot)$ and parameter vectors $\boldsymbol{\xi}_j$ are supported

by the graph G_M. After recovering Ξ via MLE, we can map the vertices of G_M onto the manifold Q based on the topology of G_M.

As the vertex set of G_Q is a Gibbs Field, it follows a Gibbs distribution. We employ the potential function

$$f_Q(\boldsymbol{r}_j, \boldsymbol{r}_i) = \sum_{\boldsymbol{x}_k \in \mathcal{X}} \alpha_{i,j} \exp\left(-\frac{1}{T}\, \mathrm{d}_M(\boldsymbol{x}_k, \boldsymbol{\xi}_j)^2\right), \tag{3.27}$$

where $\mathrm{d}_M(\cdot)^2$ is the squared geodesic distance on manifold M, $\alpha_{i,j}$ are pairwise mixture weights, and the temperature T controls the sharpness of the distribution. This gives us the likelihood

$$P(\boldsymbol{r}_i | C_{\boldsymbol{r}_i}, \mathcal{X}) = \frac{1}{Z_Q} \prod_{\boldsymbol{r}_j \in C_{\boldsymbol{r}_i}} \sum_{\boldsymbol{x} \in \mathcal{X}} \alpha_{i,j} \exp\left(-\frac{1}{T}\, \mathrm{d}_M(\boldsymbol{x}_k, \boldsymbol{\xi}_j)^2\right). \tag{3.28}$$

We view the parameters Ξ as a set of variables to be estimated and each mixture weight $\alpha_{i,j}$ as the posterior probabilities that the vertex $\boldsymbol{r}_i$ belongs to the j^{th} component of the mixture. This mixture model can be tackled by the EM algorithm. EM is used to recover maximum likelihood solutions to problems involving hidden data. It is an iterative algorithm, where in each iteration s, both the E-step (for expectation, or the estimation of posterior probabilities) and the M-step (for maximization of the expected log-likelihood) is performed. In the M-step, we maximize the expected log-likelihood with respect to the parameter vectors,

$$\boldsymbol{\xi}_j^{(s+1)} = \frac{\sum\limits_{\substack{\boldsymbol{x}_k \in \mathcal{X} \\ \boldsymbol{r}_i \in C_j}} \alpha_{i,j}^{(s)} \exp\left(-\frac{1}{T}\, \mathrm{d}_M\left(\boldsymbol{x}_k, \boldsymbol{\xi}_j^{(s)}\right)^2\right) \boldsymbol{x}_k}{\sum\limits_{\substack{\boldsymbol{x}_k \in \mathcal{X} \\ \boldsymbol{r}_i \in C_j}} \alpha_{i,j}^{(s)} \exp\left(-\frac{1}{T}\, \mathrm{d}_M\left(\boldsymbol{x}_k, \boldsymbol{\xi}_j^{(s)}\right)^2\right)}. \tag{3.29}$$

In the E-step, we estimate the posterior probabilities given the likelihood in Eq. 3.28,

$$\alpha_{i,j}^{(s+1)} = \frac{\tau_i \sum\limits_{\substack{\boldsymbol{x}_k \in \mathcal{X} \\ \boldsymbol{r}_j \in C_i}} \alpha_{i,j}^{(s)} \exp\left(-\frac{1}{T}\, \mathrm{d}_M\left(\boldsymbol{x}_k, \boldsymbol{\xi}_j^{(s)}\right)^2\right)}{\sum\limits_{\boldsymbol{r}_l} \tau_l \sum\limits_{\substack{\boldsymbol{x}_k \in \mathcal{X} \\ \boldsymbol{r}_j \in C_i}} \alpha_{i,j}^{(s)} \exp\left(-\frac{1}{T}\, \mathrm{d}_M\left(\boldsymbol{x}_k, \boldsymbol{\xi}_j^{(s)}\right)^2\right)}, \tag{3.30}$$

where τ_i is the posterior for each of the clique sets in the graph, given by

$$\tau_i = \frac{\sum_{\boldsymbol{r}_k \in C_i} \alpha_{k,i}^{(s)}}{\sum\limits_{\substack{\boldsymbol{r}_l \in \mathcal{R} \\ \boldsymbol{r}_k \in C_i}} \alpha_{l,k}^{(s)}}. \tag{3.31}$$

Effectively, this describes an unsupervised learning approach where the topology of graph G_Q constraints the interference process through its clique sets, and the parameters $\boldsymbol{\xi}_i$ govern the kernel function $\kappa(\cdot)$.

Energy Functions

If we revisit the potential function in Eq. 3.27, we note that it corresponds to a Gibbs measure. This accounts for a Boltzmann distribution [Kind 80], whose energy is the squared geodesic distance $\mathrm{d}_M(\cdot)^2$. This can be easily seen when we rewrite Eq. 3.27 as

$$f_Q(\boldsymbol{r}_j, \boldsymbol{r}_i) = \alpha_{i,j} \sum_{x \in \mathcal{X}} \exp\left(-\frac{1}{T} \mathrm{E}_{\xi_j}(\boldsymbol{x}_k)\right) , \tag{3.32}$$

where $\mathrm{E}_{\xi_j}(\boldsymbol{x}_k) = \mathrm{d}_M(\boldsymbol{x}_k, \boldsymbol{\xi}_j)^2$ is the energy of $\boldsymbol{x}_k$ with respect to $\boldsymbol{\xi}_j$ and $\alpha_{i,j}$ is a function of vertex pair $\boldsymbol{r}_i$, $\boldsymbol{r}_j$. Furthermore, in Eq. 3.27 (for instance), it is straightforward to set the mixture weight $\alpha_{i,j} = \eta_i \eta_j$, resulting in $f_Q(\boldsymbol{r}_j, \boldsymbol{r}_i) = \eta_i \eta_j \kappa(\boldsymbol{x}_k, \boldsymbol{\xi}_j)$ such that

$$\kappa(\boldsymbol{x}_k, \boldsymbol{\xi}_j) = \exp\left(-\frac{1}{T} \mathrm{d}_M(\boldsymbol{x}_k, \boldsymbol{\xi}_j)^2\right) . \tag{3.33}$$

This treatment is consistent with that commonly given to priors in Bayesian inference on multivariate mixtures [McLa 08] and opens up the possibility of using other kernels such as a uniform kernel or robust estimators [Hube 81].

3.3.2 Relationship to SOM

We now discuss how this approach is related to the SOM concept. Consider the case where the weights $\alpha_{i,j}$ do not need to be estimated but can be computed from the vertex set $\mathcal{X}$. In this case, the interference process is reminiscent of a SOM. Consider a sampling process in M and set

$$\alpha_{i,j} = [\boldsymbol{r}_j \sim \rho]\, \mathrm{h}\left(\frac{1}{T} \mathrm{d}_Q(\boldsymbol{r}_i, \boldsymbol{r}_j)^2\right) , \tag{3.34}$$

where $\mathrm{h}(\cdot)$ is a real-valued positive function and $\mathrm{d}_Q(\cdot)$ is the geodesic distance on Q, which can be computed through the constructed G_Q. As before, T is the temperature of the system and $[\boldsymbol{r}_j \sim \rho]$ is unity for the vertex $\boldsymbol{r}_j$ corresponding to the parameter variable $\boldsymbol{\xi}_j$ whose distance to $\boldsymbol{x}_k$ is minimal, i. e. the best-matching unit ρ, and zero otherwise.

Now consider the case where the energy function $\mathrm{E}_{\xi_j}(\boldsymbol{x}_k)$ is constant. Then in the sampling process, for each input sample $\boldsymbol{x}_k$, the update strategy in Eq. 3.30 will draw $\boldsymbol{\xi}_j^{(s+1)}$ towards the vector $\boldsymbol{\xi}_\rho^{(s)}$ (corresponding to vertex ρ in G_Q) which is closest to the sample $\boldsymbol{x}_k$. The resulting scheme is also consistent with the notion that, following the "cooling" schedule, the influence of the geodesic distance upon the update is decreasing. A decrease in the neighborhood influence of the SOM, $\sigma(s)$ in Eq. 3.13, can be replicated by reducing the number of adjacent neighbors, and therefore the size of each vector's clique set, in subsequent iterations s. Likewise the learning rate $\alpha(s)$ in Eq. 3.13 corresponds to temperature T in Eq. 3.34.

It is noteworthy that Generative Topographic Mapping (GTM) by Bishop et al. is a probabilistic manifold learning algorithm that can also be seen as a probabilistic

version of the SOM [Bish 98, Bish 06, Chapter 12]. In contrast to the SOM, in which the vectors from the high-dimensional space are projected into a discretized, low dimensional space, in GTM, the data in the high-dimensional space is generated from the low-dimensional nodes. GTM describes the data probability distribution in the high-dimensional space using a mixture of Gaussians embedded in the high-dimensional manifold. Each Gaussian is generated by non-linear transformations from the grid nodes in the latent space. The SOM and our proposed method operate on a set of model vectors $\boldsymbol{m}$ or parameters ξ instead, whereas other than in the SOM, the parameters ξ are kernel variables. In practice, the results on real world data can be quite different between SOM and GTM [Cott 16]. In our early experiments for a single application, we obtained quite similar results with GTM and SOM, however, the GTM took twice as long to compute at that time.

3.4 Experimental Results

In this section we show the viability of the SOM for the task of dimensionality reduction and manifold representation, as well as the performance gains achieved by our adaptations of the SOM.

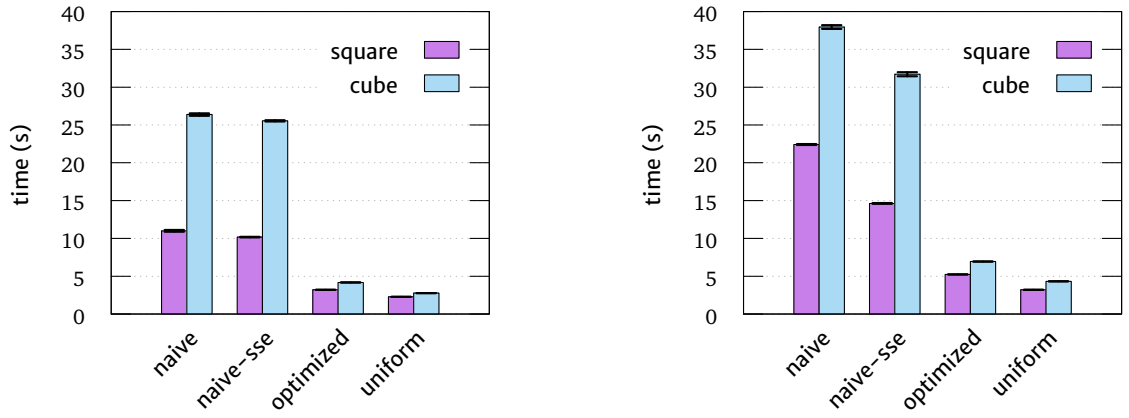
Section 3.4.1 – 3.4.3 measure computational performance and general quality of the generated maps. In these experiments, a SOM is trained first out of competition to remove data loading effects. Then, ten SOMs are trained with an individually seeded random number generator to show any variability caused by random initialization and sample drawing. Reported wall clock execution times refer to running the algorithm in question on an Intel Core i5-6600 CPU with four cores. All algorithms are implemented in C++.

In Section 3.4.4, we will use classification as an example for possible performance improvements induced by our semi-supervised training and ranked BMU lookup techniques.

3.4.1 Computational Performance in Training

In this thesis, we will present several algorithms that aid interactive analysis of an unseen multispectral or hyperspectral image. Training a new SOM on the image is a mandatory first step of these algorithms. This is why a quick training process is of high importance. We evaluate the computational performance gains effected by the changes detailed in Section 3.2.1. For this, we compare four configurations, denoted as `naive`, `naive-sse`, `optimized`, and `uniform`. The `naive` algorithm is our initial reference implementation, which includes the breadth-first traversal to prune neighborhood updates early. Our first adaptation, accelerating the computation of distances via vector instructions, is referred to as `naive-sse`. A new implementation that includes the fast distance computation and exploits the radial symmetry of $h_{c,k}$ results in the algorithm `optimized`. Finally, in `uniform`, the Gauss kernel is replaced by a uniform kernel (see Eq. 3.14).

We chose the images *Fake and Real Peppers* and *Indian Pines* for this test, as their respective band count represents two commonly found spectral resolutions ($n_D = 31$ in the visible range, and $n_D = 200$ in the visible to infrared range, respectively).



(a) *Fake and Real Peppers*, $n_D = 31$ **(b)** *Indian Pines*, $n_D = 200$

Figure 3.7: Mean training times with standard deviation as error bars for different SOM algorithms.

A higher band count leads to longer training times due to more expensive vector comparisons. To test for the influence of topological complexity, we test both a *square* and a *cube* topology, with $n_M = 1024$, and $n_M = 1000$, respectively. On both images, we feed $n_S = 50\,000$ samples during training.

Figure 3.7 depicts the training times measured for the various configurations tested on both images and topologies. We can see that the performance gain of optimized distance computation is higher with a larger number of bands. When this testing was done first in 2012, the difference between `naive` and `naive-ssm` was considerably stronger also for a lower band number. Either the optimization capabilities of a newer compiler, or the newer processor architecture reduce this effect. For both images, the strongest gain in performance is achieved by our new optimized kernel implementation. Replacing the Gauss kernel with a uniform kernel leads to the fastest algorithm and the time spent for training falls well below five seconds for both images and topologies, without a negative effect on training quality (see page 47). The new implementation also deals considerably better with the higher-dimensional cube topology, which in comparison to the square topology penalizes our training by about 20 to 35 % in the case of a uniform kernel.

We stress the efficiency of our implementation as it is important to note the viability of the SOM for an interactive setting and for speedup of other algorithms. Generally, SOMs are reported to be far slower to train. For example, in 2000, Vesanto and Alhoniemi, authors of the SOM Toolbox for Matlab [Vesa 00b], reported training times of 34 minutes (Matlab), and 22 minutes (SOM_PAK, [Koho 96]) for $n_D = 30$, $n_M = 1000$, and $n_S = 30\,000$.

3.4.2 Mapping Quality and Complexity

In this set of experiments we empirically determine parameter sets for the SOM that provide a reasonable trade-off between computational complexity and quality of the trained SOM. For consistent measurements, we apply a constant scale factor to the spectral vectors $\boldsymbol{x}$, such that all x_d fall in the interval $[0, 1]$. We contrast two

quality measures with training time. The average quantization error (AQE) is used in literature to evaluate the quality of representation that the SOM model vectors provide for the input distribution [Liu 06, Polz 04]. It is given as

$$\mathrm{AQE} = \frac{1}{n_X} \sum_{x \in \mathcal{X}} \|\boldsymbol{x} - \boldsymbol{m}_c\|_2 \,, \tag{3.35}$$

where $\boldsymbol{m}_c$ is the best-matching unit (see Eq. 3.9). We also measure the maximum absolute quantization error (MQE),

$$\mathrm{MQE} = \max_{x \in \mathcal{X}} (\|\boldsymbol{x} - \boldsymbol{m}_c\|_2) \,. \tag{3.36}$$

We deem MQE to be a helpful measure in combination with AQE, as it makes outliers visible which could indicate that the SOM is not fully representing the input distribution. An example of such a case in hyperspectral images could be specular highlights, which only account for a low percentage of overall image pixels. The error introduced by not well-representing a cluster formed by specular highlight pixels might be diminished by the overwhelming number of unaffected image pixels in AQE.

Training Sample Size

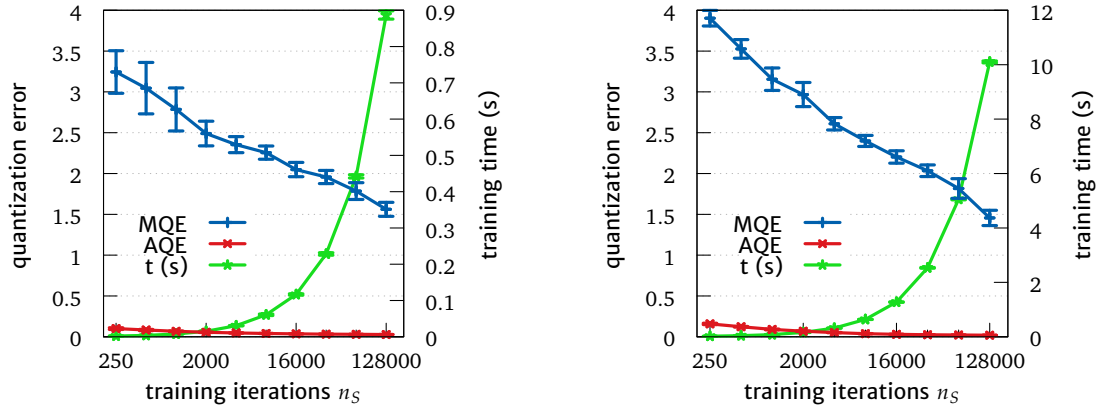
In the first experiment, we determine the most practical amount of training samples n_S for our domain. The test set consists of the images *Fake and Real Peppers* and *Indian Pines*, as well as two additional images *D.C. Mall* and *Dürer 1*. We train SOMs of both a 16×16 square lattice and a $13 \times 13 \times 13$ cube lattice, whereas the latter contains close to nine times more model vectors than the former. The algorithm coined `optimized` is used for training. The number of training samples is varied between 250 and 128 000, doubled in each test. Note that learning rate and kernel width are scaled in concern with map geometry such as each training run is complete from orientation to refinement (see Appendix C on page 167).

Figure 3.8 depicts the results for the *Fake and Real Peppers* image. As expected, training times grow linearly with the number of iterations n_S. We also deduce that, when n_S is chosen to be reasonably high, a larger map size does not necessitate a longer training. The AQE only marginally improves starting at $n_S = 16\,000$, while the MQE continues to profit from longer training times. For both it is noted that the quantization error shrinks only logarithmically with an increase of n_S. From Figure 3.9 we can deduce that the same findings hold true for images with a considerably larger number of bands (557, and 200, respectively), captured by different sensor models and band filters with respect to λ. This shows the consistency of the SOM training for multispectral and hyperspectral data.

Taking all tested images into account, we consider a choice of $n_S = 50\,000$ as a good compromise between sample representation and training times. The full results on all four images can be found in Appendix D (page 170).

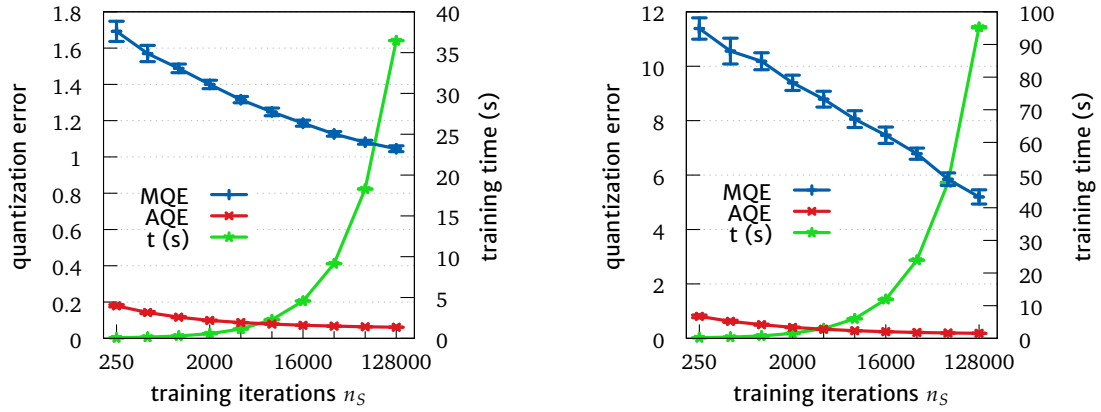
Neighborhood Function

We now consider how the choice of $\mathrm{h}_{c,k}$ may affect the mapping quality. We compare the widely used Gaussian kernel (see Eq. 3.13) with the uniform kernel (see Eq. 3.14).



(a) square, $n_M = 256$ **(b)** cube, $n_M = 2197$

Figure 3.8: Quantization error for different SOM parameters on *Fake and Real Peppers* image. n_S is plotted on a logarithmic scale.



(a) *Indian Pines*, $n_M = 2197$ **(b)** *Dürer 1*, $n_M = 2197$

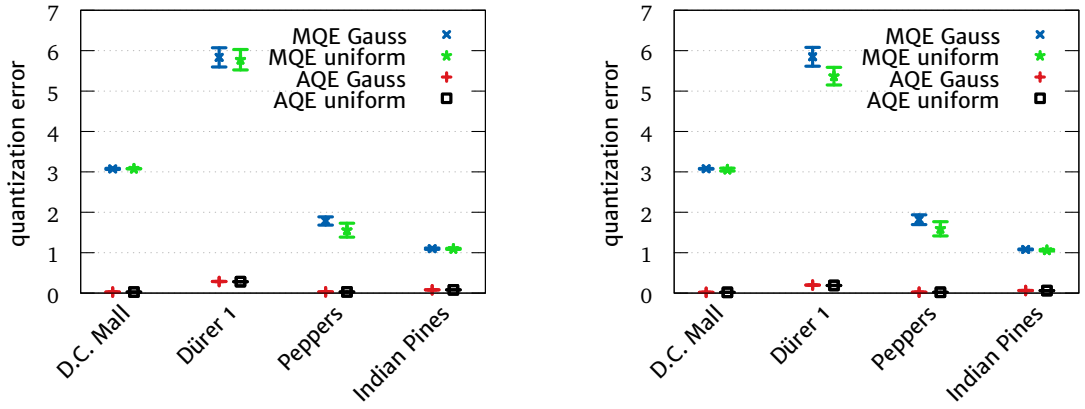
Figure 3.9: Quantization error for different SOM parameters on additional images, using a cuboid topology. n_S is plotted on a logarithmic scale.

For this, we compare the SOMs from the previous test trained with 64 000 iterations (which were trained using a Gaussian kernel) with their respective counterparts, where the only parameter changed is the kernel.

Figure 3.10 depicts the resulting quantization errors for both topologies. We observe that the change in kernel has no adverse effect on the representation quality of the map. In some cases, we even find a slight improvement. This fits our expectation that we can reduce training times by using a simpler kernel without compromising algorithmic performance. In all following experiments throughout this thesis, we will employ the uniform kernel in SOM training.

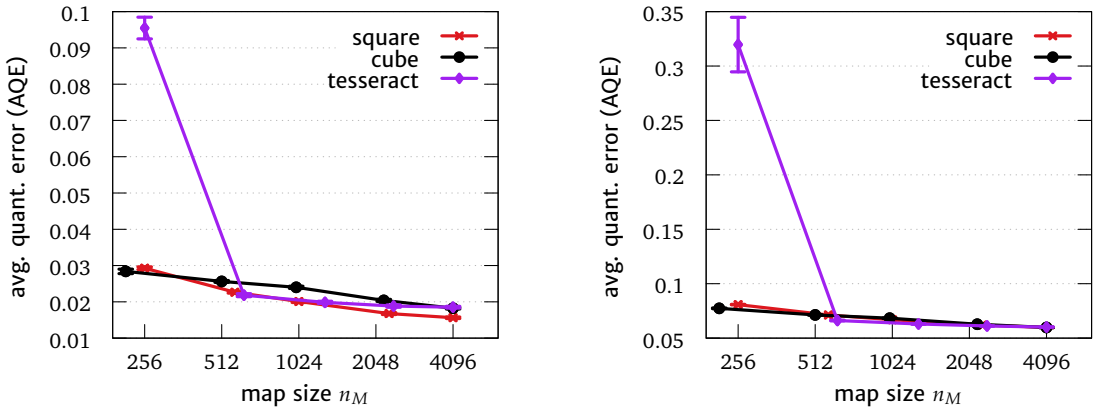
Map Size and Shape

In our applications of the SOM we are particularly interested in maps of large sizes and up to four dimensions. To test the effect of topology size and shape, we construct 15 maps, five each with a square, cube, and tesseract topology, and the count of



(a) square, $n_M = 256$ **(b)** cube, $n_M = 2197$

Figure 3.10: Quantization error for Gauss and uniform kernel on two topologies. *Fake and Real Peppers* is shortened to *Peppers*.



(a) *Fake and Real Peppers* **(b)** *Indian Pines*

Figure 3.11: Quantization error for different SOM shapes and sizes. n_M is plotted on a logarithmic scale.

model vectors varying between $\approx$ 250 and 4096. The respective sizes for each dimensionality differ slightly so they can be constructed with a fixed side length n'_M. For example, a square with $n'_M = 32$ roughly corresponds to a cube with $n'_M = 10$ and a tesseract with $n'_M = 6$.

Figure 3.11 depicts the average quantization errors for maps of various sizes and shapes. MQE was omitted for clarity. The quantization error improves with a larger number of model vectors, which is expected. However, similar to the previously observed effect of increasing the size of the training set, increasing the size of the map only leads to a logarithmic improvement of the MQE. We see two effects of the SOM's dimensionality on the quality of the quantization. One, the tesseract lattice with $n_M = 256$ fails to represent the data well, which is explained by its very limited length $n'_M = 4$ in each dimension. Two, a square SOM performs slightly better than its higher-dimensional counterparts for the *Fake and Real Peppers* image, which is not the case with other images (see two additional plots in Appendix D on page 171).

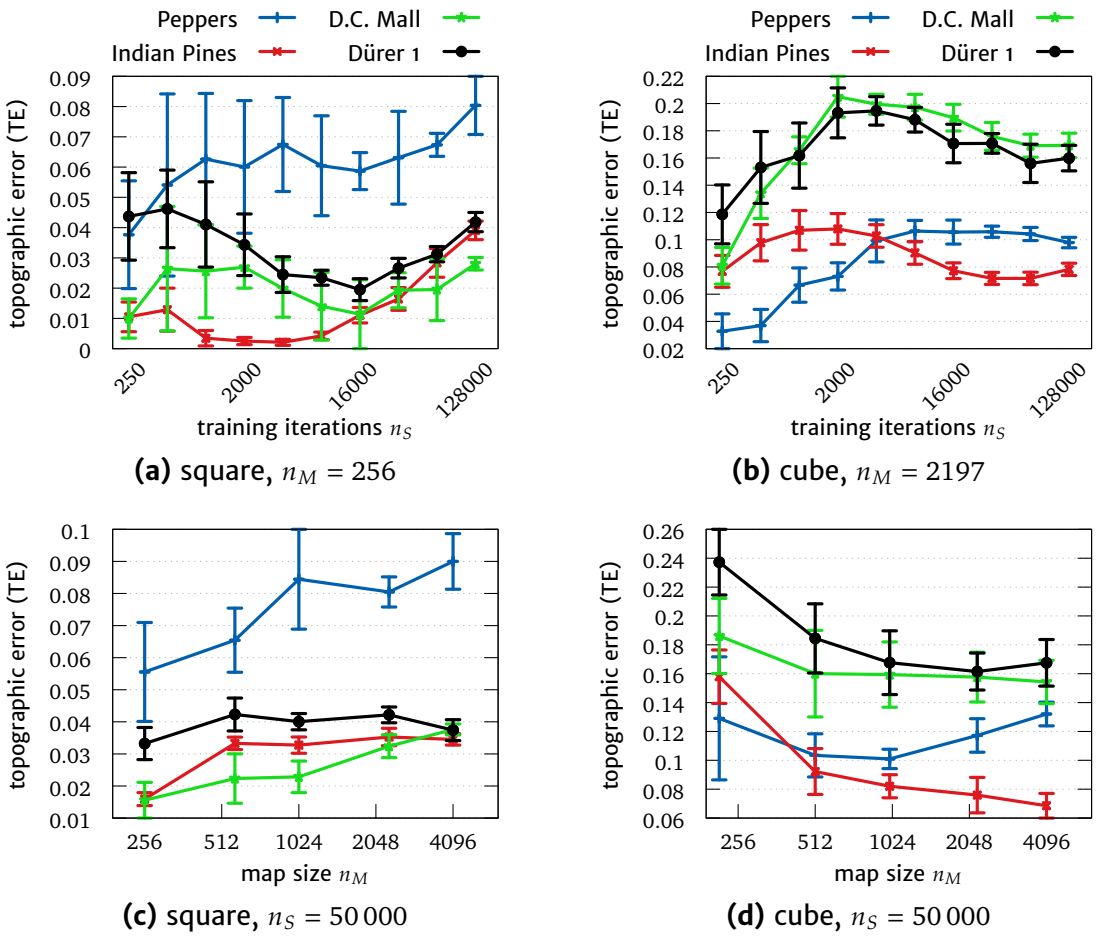
	n_M	*Peppers*	*Indian Pines*	*D.C. Mall*	*Dürer 1*
Training	256	**0.23** ± 0.00	**0.82** ± 0.00	**0.78** ± 0.01	**2.12** ± 0.00
	1024	**0.79** ± 0.01	**3.05** ± 0.01	**2.85** ± 0.01	**8.01** ± 0.01
Lookup	256	**0.48** ± 0.03	**0.16** ± 0.01	**2.56** ± 0.18	**17.25** ± 1.09
	1024	**1.88** ± 0.14	**0.58** ± 0.04	**8.01** ± 0.04	**55.14** ± 0.27

Table 3.1: SOM training and lookup times in seconds on the four test images. *Fake and Real Peppers* is shortened to *Peppers*. *Indian Pines* bears a low spatial, but high spectral resolution, while *Dürer 1* bears very high spatial and spectral resolutions.

Training and Lookup Times

After training the SOM on an image for inspection, often a lookup is needed for each pixel in a further step of our algorithms employing the SOM. At this point, we give a hint at the relation between training time and lookup time. While training complexity is a function of $n_M n_S$, lookup complexity is a function of $n_M n_X$. An important factor in the speed of SOM training is that n_S is not dependent on the image size n_X. On the other hand, in contrast to training, the lookup can be highly parallelized (using four cores in the test setup). The effect of these factors can be observed in Table 3.1. It lists the wall clock execution times for both tasks for a 2-D lattice of sizes 16×16 and 32×32 each. For all pixels, one single-BMU lookup is performed. When comparing the numbers between different images, we see the effect of n_X on the lookup, but not on the training. We also observe that image dimensionality plays an important role for both, as seen first in Figure 3.7. For large images, lookup takes considerably longer than training on the same image. As expected, both training and lookup times grow linearly with the size of the map.

The experiments regarding quantization errors and computational complexity reveal that the SOM is stable and versatile when it comes to parameter setting. For the variety of images we tested, the number of samples needed for training does not depend on the specific image. Neither does it depend on the size of the map or its dimensionality, as long as the sample set is representative of the input data. We are also free to choose the neighborhood function solely based on computational performance. Effectively, we can choose size and dimensionality based on application alone. In some applications, apart from concerns regarding the computational performance of the SOM or of an algorithm utilizing it, a specifically sparse representation of the distribution might be seeked for, calling for the training of a smaller map. In others, a larger map can help with reducing the effect of quantization and, if needed, further improve representation of the image pixels. Appendix C (page 167) provides a set of SOM configurations for various sizes and topologies that were used for these tests and will find further use throughout this thesis.



(a) square, $n_M = 256$ **(b)** cube, $n_M = 2197$

(c) square, $n_S = 50\,000$ **(d)** cube, $n_S = 50\,000$

Figure 3.12: Topographic error for different SOM parameters on the four test images. n_S and n_M are plotted on a logarithmic scale.

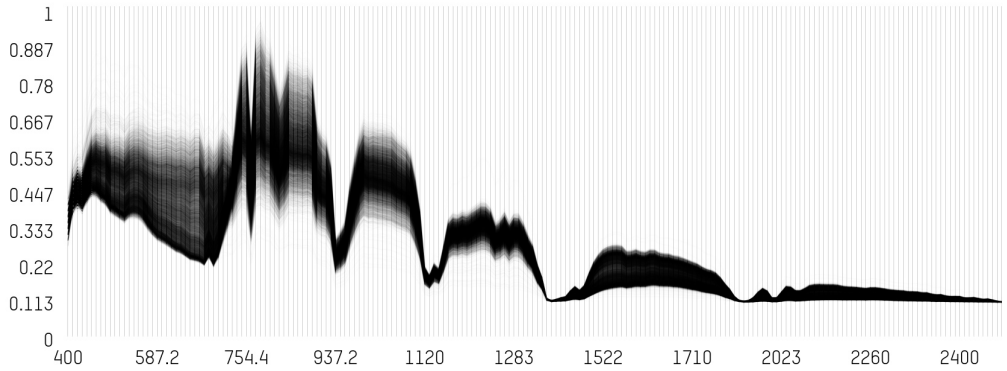
3.4.3 Distance and Topology Preservation

For testing the local consistency in the SOM topology, the topographic error (TE) is proposed in literature [Liu 06, Polz 04]. It measures the portion of pixels where BMU and second best-matching unit are not adjacent. It is given as

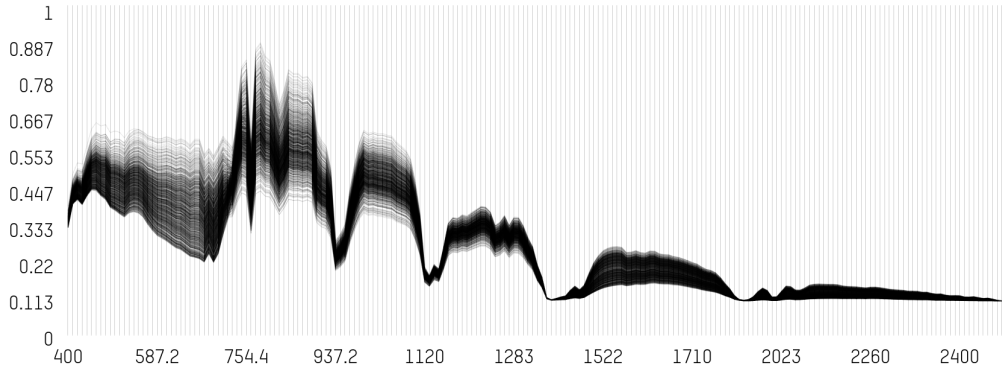
$$\mathrm{TE} = \frac{1}{n_X} \sum_{\boldsymbol{x} \in \mathcal{X}} \left[\left\| \boldsymbol{r}\left(c_1^{(\boldsymbol{x})}\right) - \boldsymbol{r}\left(c_2^{(\boldsymbol{x})}\right) \right\|_\infty > 1 \right] , \tag{3.37}$$

where $\boldsymbol{c}^{(\boldsymbol{x})}$ is given by Eq. 3.15, and the number of desired BMUs $n_C = 2$.

Figure 3.12 shows the TE as measured in the previous experiments on training time and map size. In our experiments, the TE appears not to be a very telling measure. Other than in the measurements of AQE and MQE, it behaves differently for different images. It is counterintuitive that it may even increase for a SOM with longer training. Our explanation for this observation is that when the SOM learns more detailed differentiations between input samples from the original distribution, their representation in the map topology may grow. It is not straightforward to define how discriminant the SOM ought to be in the sense that some of this growth can be attributed to the map reflecting variations caused by noise. This phenomenon



(a) input spectral distribution



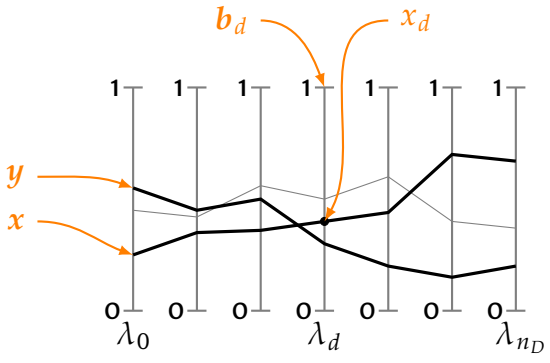
(b) SOM model vector distribution

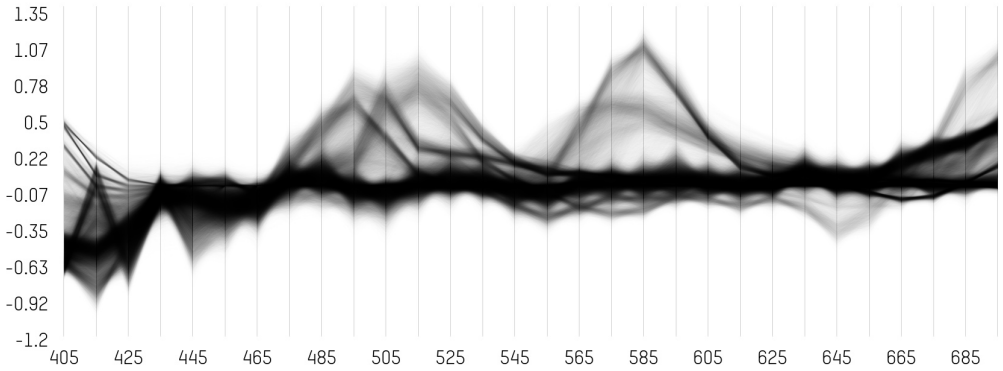
Figure 3.13: Data distributions obtained for *Indian Pines* image.

will be revisited in Section 4.1.2. We qualitatively evaluate the topology preservation instead. For this, we compare the distributions of $\mathcal{X}$ and $\mathcal{M}$ in the high-dimensional feature space. To do so, we use the spectral distribution plots introduced in Section 5.3. We give a brief explanation of the plots here, which are based on the concept of Parallel Coordinates [Inse 90].

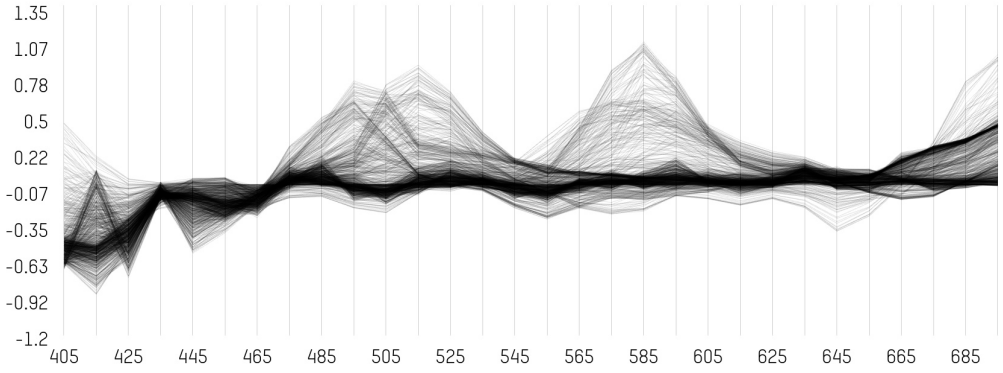
Spectral Distribution Plots

An array of n_D parallel vertical lines represents the n_D spectral bands. The y-coordinate on the dth axis corresponds to a spectrum's value at band d. To display the spectral vector of a pixel $\boldsymbol{x}$, a polyline is drawn with its vertices lying on the corresponding vertical axes. The resulting display follows the layout of a plot where the x-axis would denote wavelength, and the y-axis denotes intensity. Alpha blending is used for a combined display of all data samples.





(a) input spectral gradient distribution



(b) SOM model vector distribution

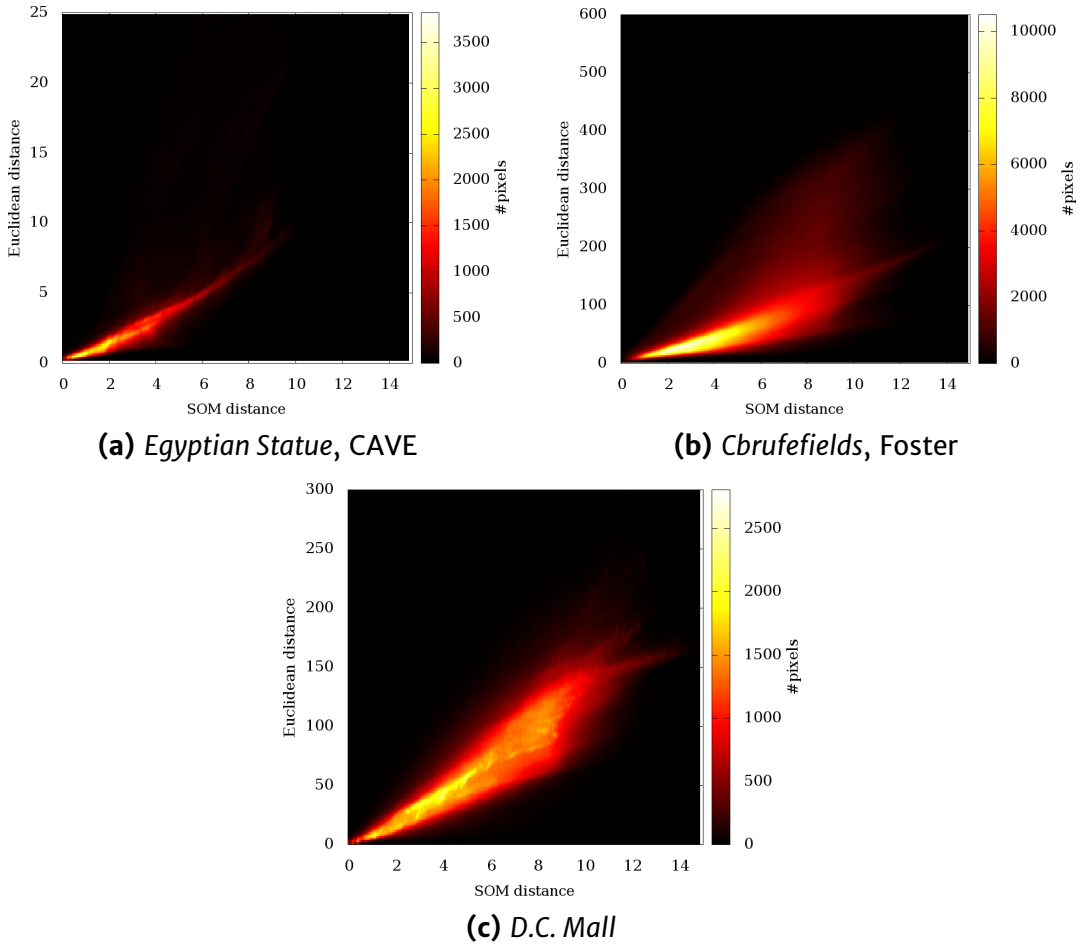
Figure 3.14: Data distributions obtained for *Fake and Real Peppers* image.

Figure 3.13 compares the spectral distribution of the *Indian Pines* image with the distribution of the SOM model vectors. A 3-D SOM with $n_M = 1000$ was trained on the image for this display. When comparing the distributions, we see that the SOM model vectors form a sparse representation of the original distribution. However, one aspect notable in close inspection is that the SOM, due to the nature of its training, is slightly smoothing out the outer edges of the distribution, therefore given the impression of a slight shrink. This effect might be most notable in the range 1500 nm to 1900 nm. It is an explanation of the MQE measuring significantly higher than the AQE. In Figure 3.14, the same comparison is made for the *Fake and Real Peppers* image, however, in this case, we trained a SOM on the spectral gradient of the image. As will be illustrated in Section 5.5.3, for this image, the spectral gradient describes the materials in the scene considerably better than the original feature space, due to its geometry invariance. We can see in Figure 3.14 that this description stays intact in the SOM representation of the distribution.

Distance Preservation

A good topology preservation in the SOM implies a good distance preservation, which is a likewise favorable property. Distances are preserved through the mapping $\Gamma : \mathbb{R}^{n_D} \rightarrow \mathbb{R}^{n_R}$ using a SOM,

$$\Gamma(\boldsymbol{x}) = \boldsymbol{r}' \,, \tag{3.38}$$



(a) *Egyptian Statue*, CAVE

(b) *Cbrufefields*, Foster

(c) *D.C. Mall*

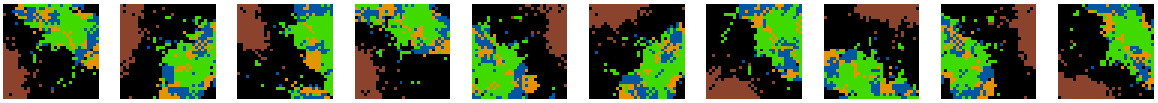
Figure 3.15: Scatter-plot histograms for pairwise distances $\|x - y\|_2$, $\|\Gamma(x) - \Gamma(y)\|_2$.

where the representative location r' is given by Eq. 3.16, with $n_C = 5$ and w given by Eq. 3.19 (see pages 37, 38). In this experiment, we use a cuboid SOM, so $n_R = 3$. The underlying assumption is that, as the model vectors of the SOM are ordered in the map according to the topology of the learned manifold, a distance in the map is a low dimensional approximation, or preservation, of a distance on the manifold.

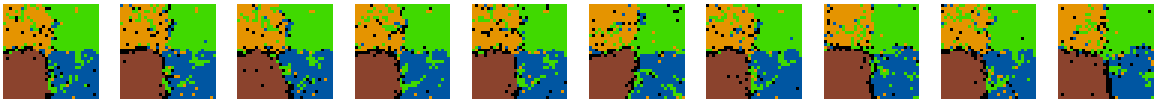
To evaluate this property, for each pixel x in the image, 40 other pixels y are randomly selected. For each of those, we calculate the distance pair $\|x - y\|_2$ and $\|\Gamma(x) - \Gamma(y)\|_2$. Figure 3.15 shows scatter-plot histograms for the pairwise relationships on three images representing multispectral lab scene, multispectral natural scene, and hyperspectral remote sensing capture. The plots reveal that small distances exhibit a high correlation for the three images. Larger distances diverge from a strictly linear relationship. This is an expected result, as the Euclidean metric may fail to approximate non-local distances within the underlying manifold. In this case, the pairwise distances increase.

3.4.4 Ranked BMU Lookup and Semi-supervised Training

We test our assumptions that ranked BMU lookup and semi-supervised training improve the performance for SOM-based analysis with a classification task. The



(a) Unsupervised training



(b) Semi-supervised training

Figure 3.16: Maps of class labels for SOM neurons after training on *Indian Pines* image with four classes from Figure 2.5. Black neurons vote for no class.

Indian Pines remote sensing image comes with 16 classes for which ground-truth labels are available, as depicted in Figure 2.5 (page 13). Of the 220 available bands, 200 were kept as described in Section 2.1.3 and transformed via $\mathrm{N}(\boldsymbol{x})$ according to Eq. 2.2. We perform a ten-fold cross validation where in each fold, 10 % of the samples in each class are used for testing. Training is done on the other 90 % of samples in the supervised orientation phase, and the rest of the image in the unsupervised refinement phase. The weight functions ω_l controlling the magnetism in the supervised orientation phase are obtained via Eq. 3.22. In the case of unsupervised training, the image data is used without labels.

After training, we go through all training samples $\mathcal{S}$ to determine which neuron in the SOM represents each sample. During the classification, each neuron $\boldsymbol{m}_k$ will vote for the classes of the samples it represents. This means, it may vote for several classes,

$$v_l^{(k)} = \sum_{\boldsymbol{x}\in\mathcal{S}} [c^{(\boldsymbol{x})} = k][\mathcal{L}(\boldsymbol{x}) = l]\,, \tag{3.39}$$

where $c^{(\boldsymbol{x})}$ is obtained by Eq. 3.9 and $\mathcal{L}(\boldsymbol{x})$ is the class label of training sample $\boldsymbol{x}$. $\boldsymbol{v}^{(k)}$ is a vector that holds the number of votes per class label for the neuron at index k. The prediction for each test sample $\boldsymbol{y}$ is

$$\mathrm{a}(\boldsymbol{y}) = \arg\max_l v_l^{(k)}, k = c^{(\boldsymbol{y})}\,, \tag{3.40}$$

which is the class label that obtains most votes from the best-matching unit.

For illustration purposes, we first perform the classification with four classes and a 2-D SOM. The four classes are selected due to their prominence in the ground truth labeling with many available samples: *Soybean min till*–2455, *Corn no till*–1428, *Woods*–1265, *Corn min till*–830. Yet we see a high discrepancy in available samples, with approx. three times more training samples being available for *Soybean min till* than for *Corn min till*. We train one SOM unsupervised, another semi-supervised. Due to the cross-validation, we obtain ten SOMs in each case. Figure 3.16 shows the SOM topologies, colored by the class label obtained through $\mathrm{a}(\boldsymbol{m}_k)$ (Eq. 3.40) for each neuron $\boldsymbol{m}_k$ at its location $\boldsymbol{r}^{(k)}$. We observe that in the unsupervised case, SOM orientation changes between runs, but the shape of the classes in the SOM is quite

similar. It is also evident, that many neurons do not represent any labeled samples at all. This is expected, given the various other materials present in the scene that are represented by these neurons. We get a different picture in the semi-supervised case: Here, the classes are mostly represented by neurons being located in their dedicated quadrant of the SOM topology. Furthermore, the representation of these classes is more even, and only a small number of neurons does not represent samples from these classes. It shows that the supervised orientation phase works as expected.

We now also incorporate ranked BMU lookup in the prediction step. We reason that higher accuracy can result from a richer understanding of how a sample is represented in the map. The ranked BMU prediction for each test sample $\boldsymbol{y}$ is

$$\mathrm{a}'(\boldsymbol{y}) = \arg\max_{l} \left(\sum_{j=1}^{n_C} w_j v_l^{(k_j)} \right), \boldsymbol{k} = \boldsymbol{c}^{(\boldsymbol{y})}, \tag{3.41}$$

which is the class label that obtains most votes from all best-matching units (recall Eq. 3.15), whereas the contribution of each unit is weighted in a similar vein as in Eq. 3.16. $\boldsymbol{w}$ is obtained via Eq. 3.19. We effectively obtain a classifier in the spirit of k Nearest Neighbors (kNN), whereas SOM units compactly represent the original samples.

For the classification task on all 16 classes, we benchmark several variants to determine the impact of each algorithmic change. The baseline is a simple unsupervised SOM with a tesseract topology and $n'_M = 6$. We also test a bigger SOM with $n'_M = 8$. The single-BMU lookup (Eq. 3.40) is contrasted with multi-BMU lookup (Eq. 3.41) and a varying number of BMUs n_C. Furthermore, we test the effect of our rank weights against a flat weighting, $\forall j : w_j = 1$. Finally, all configurations are also tested with semi-supervised training.

Table 3.2 lists overall accuracy (OA) and average accuracy (AA) of all tested variants. While OA is the percentage of correctly classified pixels, AA is the mean of class-specific accuracies. AA is typically lower on this test image due to the high discrepancy of available samples between classes. For each row in the table, a bullet point denotes the BMU lookup configuration (whereas $n_C = 1$ denotes single-BMU lookup) and check marks denote configuration variants to the baseline configuration. Most notable in the results is that on this particular task, our efforts to improve the SOM performance work best when combined: A notably bigger map, feasible due to our efficient version of the algorithm, gives the best results. Yet, size alone does not beat the combination of ranked BMU lookup and supervised training. A high value of n_C leads to the best results in the field when used with the rank weights, and plays a particular role in improving the AA. However, it can show a detrimental effect when used in a flat kNN scheme, leading to the worst results in the field in combination with a smaller SOM. The confusion matrices for the baseline configuration with $n_C = 1$ and the best-performing configuration can be found in Appendix D on pages 172–173.

In this experiment we showed a proof-of-concept for both the feasibility of our custom SOM-based classification method outlined in these experiments, as well as for the positive effect of our novel SOM algorithms in training and lookup. Note that these preliminary results are not generally competitive with the huge body of

n_C =	1	5	10	15	big SOM	ranked BMU	supervised	OA	AA
				•				**67.0 %**	54.6 %
			•					**69.3 %**	59.6 %
				•			✓	**69.4 %**	66.5 %
			•				✓	**73.0 %**	71.8 %
	•				✓			**73.4 %**	71.8 %
		•						**74.2 %**	69.8 %
				•	✓			**74.9 %**	70.6 %
	•							**75.1 %**	73.5 %
	•				✓		✓	**76.4 %**	58.3 %
			•		✓			**77.0 %**	74.8 %
		•					✓	**77.1 %**	74.8 %
		•				✓		**77.3 %**	75.4 %
			•			✓		**77.4 %**	75.4 %
				•		✓		**77.4 %**	75.4 %
		•			✓			**78.2 %**	78.0 %
				•	✓		✓	**78.6 %**	75.3 %
	•						✓	**78.6 %**	69.3 %
		•			✓	✓		**79.3 %**	78.7 %
			•		✓	✓		**79.6 %**	78.9 %
				•	✓	✓		**79.7 %**	78.9 %
			•		✓		✓	**80.3 %**	73.6 %
		•				✓	✓	**81.0 %**	78.3 %
			•			✓	✓	**81.1 %**	80.2 %
				•		✓	✓	**81.2 %**	80.5 %
		•			✓		✓	**81.6 %**	72.0 %
		•			✓	✓	✓	**83.4 %**	73.8 %
			•		✓	✓	✓	**84.1 %**	78.0 %
				•	✓	✓	✓	**84.3 %**	81.3 %

Table 3.2: Overall accuracy in classification on *Indian Pines* image. Measured are overall accuracy (OA) and average accuracy (AA). Methods are ordered by OA.

existing, considerably tuned methods based on kernelized SVM classifiers [Moun 11]. Tarabalka et al. report an overall accuracy of 78.76 % (AA: 69.66 %) for 16 classes in a comparison with other methods that incorporate spatial features [Tara 09]. They use only 10 % of samples for training. Camps-Valls and Bruzzone report an overall accuracy of 94.44 % for nine classes [Camp 05], stripping seven classes from the test due to low number of ground-truth samples.

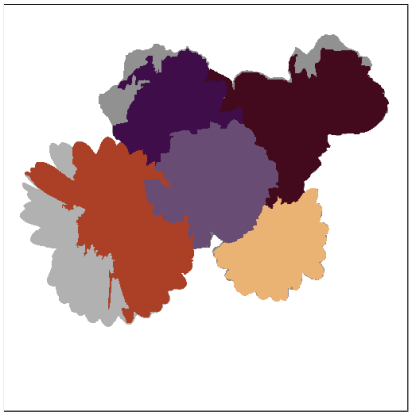
3.5 Discussion

In this chapter, we discussed how dimensionality reduction is a powerful, and often necessary, pre-processing step to combat the issues we face when analyzing hyperspectral data in the original, high dimensional feature space. Various methods for dimensionality reduction are established practice, e. g. PCA that provides a fast, easy to implement and understand, linear projection. However, it is hard to find a method that can preserve non-linear relationships in the data manifold, yet is computationally efficient enough for interactive analysis of a hyperspectral image.

The Self-organizing Map is a proven versatile tool that performs a fast, quantized manifold learning for efficient dimensionality reduction. In our experiments, we showed the reliability of the SOM when applied on both multispectral and hyperspectral input. We then took a step further in improving the SOM's performance both in terms of computational efficiency and in algorithmic output. Our ranked BMU lookup and semi-supervised training are shown to significantly boost performance on an example classification task. We also introduced a new probabilistic manifold learning approach that closely resembles the SOM, but is formulated through the interaction of the observed data samples with a random field.

We argue that many multispectral and hyperspectral analysis tasks can benefit from manifold learning. The SOM and our probabilistic interpretation within the EM framework are only two approaches to tackle this need and serve as a proof-of-concept. In the remainder of this thesis we will see several applications that benefit from this technique. Figure 3.17 depicts four algorithmic results where our custom SOM manifold learning was applied to the the multispectral image *Flowers*, and the corresponding sections where they will be introduced and discussed in detail.

(a) Edge Detection, Section 4.1



(b) Supervised Segmentation, Section 4.3

(c) False-color Visualization, Section 5.2

(d) Fuzzy Clustering, Section 4.2

Figure 3.17: Example applications of hyperspectral SOM on the *Flowers* image depicted in Figure 3.2 (see page 33).

Chapter 4

Image Analysis

In this work, we develop a new software framework for interactive analysis of multispectral and hyperspectral data. While visualization techniques provide for a fruitful qualitative analysis of the data, algorithms that analyze the image data provide quantitative input to the user. Both go hand in hand, e. g. quantitative cluster analysis can provide affiliation of image pixels to certain clusters, which is then picked up by the visualization algorithm to present a comparative display. On the other hand, visualization is often needed to assess the plausibility or quality of an analysis method when ground-truth information is unavailable.

In this chapter, we discuss three of the most-relevant image processing problems in many hyperspectral applications. The term *edge detection* describes the task of finding local discontinuities in image brightness, which are an important feature for scene understanding. They may help to find borders of objects in a scene, or points of interests (e. g., corners) which are relevant for key point-based algorithms. However, in an image with multiple components, brightness is often not a sufficient criterion to find relevant edges, a problem that magnifies when going into hyperspectral data. We will immerse ourselves in this topic and introduce a new solution to this problem in the first section.

Scene understanding can also be initiated from the opposite direction. Based on the richness of a multispectral pixel, global *clustering* algorithms may disregard any knowledge about the spatial layout of the image. Here, clusters in the spectral distributions are identified, which eventually leads to a segmentation of the whole image at once. An efficient computation of such a clustering is typically based on prior knowledge, e. g. the number of clusters or parameters of their shape. More versatile algorithms can be too computationally expensive for an interactive setting. In the second section of this chapter, we concentrate on such methods and propose more efficient variants.

The third image processing problem we discuss is related to both clustering and edge detection. *Supervised segmentation* works with the spatial layout of the image and a set of user-provided pixels for which the correct assignment is known. The algorithm then finds all missing assignments by deriving a locally consistent segmentation. As, to the best of our knowledge, there is no existing supervised segmentation method designed for the hyperspectral domain, we choose an existing

algorithm that we find most suitable for our task and adapt it to multispectral and hyperspectral images.

4.1 Edge Detection

Edge detection is a well-understood preprocessing step for computer vision applications. For example, the Canny operator dates back to 1986 [Cann 86], but is still widely in use. Canny and other common methods are based on the concept of a gradient in the spatial domain and by themselves are limited to monochromatic images. Adapting them to multispectral or hyperspectral images, where each pixel holds a high-dimensional spectral vector is a difficult problem. A common methodology is to depart from the concept of an image gradient and instead employ vector order statistics. Two established methods for hyperspectral edge detection are based on vector ordering: The Robust Color Morphological Gradient (RCMG) [Evan 06], which was designed for color images, but can be extended to the hyperspectral domain, and the method by Toivanen et al. [Toiv 03], which was designed for hyperspectral images.

In this section, we will first give an overview of related work with a focus on a hyperspectral Laplace filter, RCMG as the current state-of-the-art for color edge detection, and the method of Toivanen et al. which is a precursor to our work. We will then introduce a new manifold-learning based pseudometric for edge detection that can be used in combination with RCMG. We show how it overcomes the major weaknesses of established methods.

4.1.1 Related Work

A monochromatic image can be seen as a discrete two-dimensional image function $\mathrm{I}(l, m)$ that, for each pixel coordinate l, m, returns a non-negative value $x_{(l,m)}$, the pixel intensity. Edge detection is then often performed by analyzing the first order or second order derivatives of this function. In our case, $\mathrm{I}(l, m)$ returns a spectral vector $\boldsymbol{x}_{(l,m)}$. Extending second order methods for this case results in complex algorithms [Di Z 86, Cuma 91, Bakk 02]. Also, such gradient-based methods may fail to detect edges in the case of opposing gradients present in different spectral bands [Drew 94, Trah 93]. Bakker and Schmidt substitute the discrete derivation with a spectral dissimilarity measure [Bakk 02]. The discrete Laplace operator computes the sum of second order derivatives and is suitable for edge detection [Gonz 08, Chapter 3]. It is expressed by the Laplace filter kernel. However, the isotropy of the Laplace filter kernel can be improved using

$$\boldsymbol{L} = 2\begin{bmatrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{bmatrix} + \begin{bmatrix}1 & 0 & 1\\ 0 & -4 & 0\\ 1 & 0 & 1\end{bmatrix} = \begin{bmatrix}1 & 2 & 1\\ 2 & -12 & 2\\ 1 & 2 & 1\end{bmatrix}, \tag{4.1}$$

which is a weighted combination of the Laplace filter kernel with its diagonal counterpart. Bakker and Schmidt call this the omnidirectional Laplace operator [Bakk 02].

To apply this filter on spectral vectors, it is replicated with a sum of pairwise dissimilarities,

$$\begin{aligned} \mathrm{L}(l,m) = \frac{1}{12}\Big(& \mathrm{d}(\boldsymbol{x}_{(l-1,m-1)}, \boldsymbol{x}_{(l,m)}) + 2\,\mathrm{d}(\boldsymbol{x}_{(l,m-1)}, \boldsymbol{x}_{(l,m)}) \\ & + \mathrm{d}(\boldsymbol{x}_{(l+1,m-1)}, \boldsymbol{x}_{(l,m)}) + 2\,\mathrm{d}(\boldsymbol{x}_{(l-1,m)}, \boldsymbol{x}_{(l,m)}) \\ & + 2\,\mathrm{d}(\boldsymbol{x}_{(l+1,m)}, \boldsymbol{x}_{(l,m)}) + \mathrm{d}(\boldsymbol{x}_{(l-1,m+1)}, \boldsymbol{x}_{(l,m)}) \\ & + 2\,\mathrm{d}(\boldsymbol{x}_{(l,m+1)}, \boldsymbol{x}_{(l,m)}) + \mathrm{d}(\boldsymbol{x}_{(l+1,m+1)}, \boldsymbol{x}_{(l,m)})\Big)\,, \end{aligned} \tag{4.2}$$

where for $\mathrm{d}(\cdot,\cdot)$, Bakker and Schmidt employ the spectral angle (SA, Eq. 2.11). We may use other measures for $\mathrm{d}(\cdot,\cdot)$ according to our needs. Recall popular measures from Section 2.4 (pp. 20–23). The quality of the output then fully depends on the chosen measure's ability to capture relevant discontinuities, which is often limited. Note that this method only provides a magnitude, but not the direction of an edge.

Vector Ordering and RCMG

Another popular approach to edge detection for images of multiple bands is vector order statistics [Trah 93]. Such ordering-based methods determine edge probability by analyzing the order of all pixels in the local neighborhood. An ordering method that is most used in edge detection and filtering of multispectral images is reduced or aggregate ordering (R-ordering) [Barn 76]. Here, spectra are ordered based on distance calculations, e. g. the aggregate distance between the spectrum and all other observed spectra, or other spectra observed within a defined neighborhood [Trah 93]. It has been shown that like gradient-based methods, R-ordering edge detection is also prone to missing edges [Toiv 03]. Pixels holding different values may be mapped to the same scalar.

The Robust Color Morphological Gradient (RCMG) by Evans and Lin might be seen as the latest successor for R-ordering-based gradient calculation, and combines R-ordering with a structuring element as follows [Evan 06]. The classic morphological gradient operator for grayscale images is the difference between a dilation and an erosion [Rive 93], and can be expressed as

$$\begin{aligned} \nabla(f) &= \max_{x\in G}\big(f(x)\big) - \min_{y\in G}\big(f(y)\big) \\ &= \max\big(f(x) - f(y)\big)\ \ \forall x, y \in G\,, \end{aligned} \tag{4.3}$$

where G is a structuring element. The Color Morphological Gradient (CMG) is then given as

$$\mathrm{CMG} = \max_{x,y\in\mathcal{S}}\big(\mathrm{d}(\boldsymbol{x},\boldsymbol{y})\big)\,, \tag{4.4}$$

where $\mathcal{S}$ is the set of all pixels in G and $\mathrm{d}(\cdot,\cdot)$ may be the Euclidean distance, but see below. The RCMG is an extension to CMG that is supposed to be more robust to outliers. For this, Evans and Lin employ a clever scheme to determine a number of pixel pairs to be iteratively removed from the set, each consisting of the two pixels that are furthest apart [Evan 06]. We obtain

$$\mathrm{RCMG} = \max_{x,y\in\{\mathcal{S}-\mathcal{R}\}}\big(\mathrm{d}(\boldsymbol{x},\boldsymbol{y})\big)\,, \tag{4.5}$$

where $\mathcal{R}$ is the set of removed pairs. It has been shown that RCMG and slight variations of it perform well for color image edge detection [Nezh 11, Mitt 12]. RCMG was designed with RGB pixels in mind, but can easily be extended for higher-dimensional vectors [Tara 10].

While the RCMG algorithm can be formulated using the Euclidean distance, the authors state that L_2 might be an inferior metric on the RGB color space. Algorithms working with color in images often operate in other, perceptionally motivated colorspaces [Coma 02]. Evans and Liu instead opt for a change in metric. They employ a pseudometric introduced by Androutsos et al. [Andr 99] that combines the Spectral Angle with the Euclidean norm as

$$\mathrm{SAL}_2(\boldsymbol{x}, \boldsymbol{y}) = 1 - \left(1 - \frac{2}{\pi} \cos^{-1}\left(\frac{\langle \boldsymbol{x}, \boldsymbol{y}\rangle}{\|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{y}\|_2}\right)\right)\left(1 - \frac{\|\boldsymbol{x} - \boldsymbol{y}\|_2}{\sqrt{3 \cdot 255^2}}\right), \tag{4.6}$$

where the normalization of the magnitude component is designed for three-band, 8-bit pixels. While theoretically, this term might be adapted to the respective band count of hyperspectral input, note that Euclidean distances tend to diminish in high-dimensional spaces, and its effect on SAL_2 would lose significance. As was noted on page 21, we typically observe a comparable or better performance with the Chebyshev distance, so for hyperspectral input, we replace SAL_2 with

$$\mathrm{SAL}_\infty(\boldsymbol{x}, \boldsymbol{y}) = 1 - \left(1 - \frac{2}{\pi} \cos^{-1}\left(\frac{\langle \boldsymbol{x}, \boldsymbol{y}\rangle}{\|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{y}\|_2}\right)\right)\left(1 - \frac{\|\boldsymbol{x} - \boldsymbol{y}\|_\infty}{I_{\max}}\right), \tag{4.7}$$

where $I_{\max}$ denotes the maximum observable intensity in a band and depends on the imaging sensor (typical values range between 2^8 and 2^{14}).

A perceived weakness of RCMG is the reliance on a structuring element for defining a strictly local ordering. Based on image region, the same pair of pixels $\boldsymbol{x}$ and $\boldsymbol{y}$ may trigger a different response. Also, as is the case with the replicated Laplace filter, RCMG output depends on the ability of SAL_∞ to highlight relevant discontinuities.

SOM-based Ordering

Toivanen et al. employ a Self-Organizing Map (SOM) to generate a global ordering of spectral vectors [Toiv 03]. The SOM algorithm used in their work matches our explanation in Section 3.1.3. With a global ordering, a one-to-one correspondence between pixel values and scalars is guaranteed. The edge probability is only determined by the adjacent pixels, invariant to regional characteristics.

For the ordering, location scalars $r \in \mathbb{N}$ or vectors $\boldsymbol{r} \in \mathbb{N}^2$ are used, obtained by a 1-D SOM (2-connected topology) or 2-D SOM (4-connected topology), respectively. While the former provides a natural way of obtaining a linear order, the latter is better suited for larger SOMs that strive to cover more the complex relationships often found in the input data. This method solves problems present in previous approaches, however it is highly dependent on a good global ordering. As such an ordering is hard, if not impossible to find for complex scenes, it suffers from edge artifacts. We now discuss this observation in more detail for both cases.

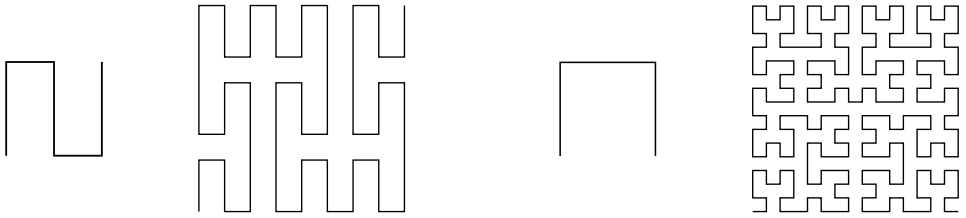
(a) image in true color **(b)** rank map R

Figure 4.1: Rank map obtained by 1-D SOM ordering on *Fake and Real Beers* image.

The 1-dimensional Case The order of spectral vectors can be easily generated if neurons of a SOM with size n_M are ordered by a scalar location index $1 \leq r_k \leq n_M$. We obtain a monochromatic image for edge detection as follows. For each pixel $\boldsymbol{x}$ of the input image, we determine $\boldsymbol{m}_c$, its best-matching unit (BMU) in the SOM (Eq. 3.9, page 31). Its map location $r^{(c)}$ is directly used as an intensity value in the rank map $\boldsymbol{R}$. The edge detector works on $\boldsymbol{R}$ which is expected to provide a strong gradient between pixels of significantly different spectral responses.

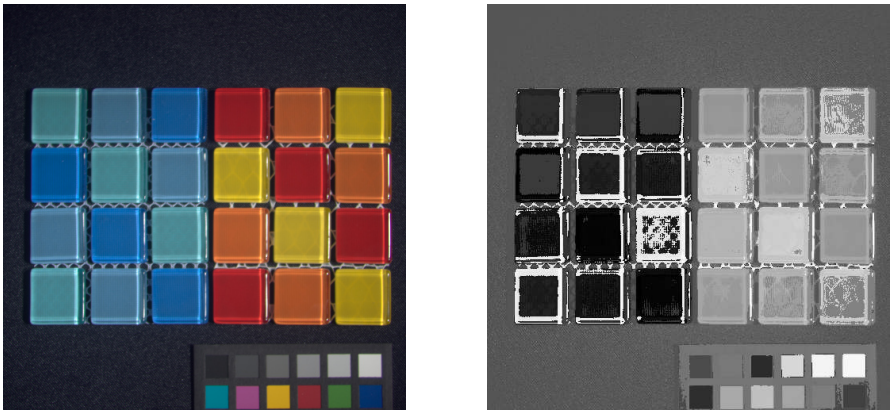
The quality of the obtained ordering depends on the SOM organization. In a well-organized SOM you see smooth transitions between spectra of adjoining neurons as well as a good global clustering of similar spectra. This means that slightly different shades of the same material observed in the scene are expected to be represented by neurons close to each other. In the 1-dimensional SOM, this is violated for images depicting more complex scenes. Each model vector can only be part of two neighborhoods that express a specific proximity relationship in the input data space. For more complex clusters of data in the input space, some proximity relationships are distorted or lost; vectors from one cluster will end up in several distinct locations of the SOM. This can result in edges introduced between pixels that are part of such a cluster. Figure 4.1 shows the *Fake and Real Beer* image from the CAVE dataset next to the rank image obtained by a 1-dimensional SOM. The SOM with $n_M = 64$ neurons was trained with 100 000 random samples from the image. As seen in the figure, it does not relate well the different reflectance effects in the scene. The rank map has outliers dominating other rank transitions. The strongest edges to be detected are false.

Due to this aforementioned limitation, the 1-dimensional SOM is not advisable in a larger scale regarding its number of neurons n_M. In principle, it would be reasonable to train a SOM with larger n_M to cover variation in the spectra in more detail and enable a more fine-grained edge detection. However, in the case of a 1-D SOM, a better coverage of detail changes in the image is traded with higher risk of false edges based on amplified location discrepancy.



(a) Peano 3×3 and 9×9 **(b)** Hilbert 2×2 and 16×16

Figure 4.2: First order and higher orders of space-filling curves. The 4th order Hilbert curve of size 16×16 is used to generate the rank map in Figure 4.3b.



(a) image in true color **(b)** rank map R from 2-D SOM

Figure 4.3: 2-D SOM ordering on *Glass Tiles* image. See Figure 4.8 on page 73 for edge detection result.

The 2-dimensional Case The problem of the 1-dimensional SOM described in the previous section lead to the choice of a higher dimensional topology. For spectral vector ordering, this introduces the need for a linearization on the 2-dimensional lattice. Toivanen et al. solve this task with space filling curves [Saga 94].

The Hilbert curve and Peano curve describe a recursive rule for traversing a 2-dimensional lattice of size $2^a \times 2^a$, and $3^a \times 3^a$, respectively ($a \in \mathbb{N}$). These curves are designed to provide a close linear index for each pair of 2-dimensional coordinates which would also be close according to their Euclidean distance. The drawback of this method lies in the portion of coordinates where this relationship analogy to the Euclidean distance is violated. Due to the recursive nature of the space filling curve, as illustrated in Figure 4.2, neurons that are adjacent in the SOM can end up with a difference in the linear index of over $\frac{4}{5}n_M$. Therefore, the topological organization of the SOM is not well covered by the resulting order and associated intensity values that are used for edge detection. This problem is significant, as can best be seen when the method is applied on another image from the same dataset, as depicted in Figure 4.3. To produce this result, a SOM with $n_M = 256$ was trained. The rank map shows inconsistent ranks for similar pixels, in some cases with huge gaps between the ranks. As a result, the edge detector cannot avoid false edge artifacts without losing valuable edge information.

Statistical Edge Detection

The Saliency-based Edge Detection in Multispectral Images (SEDMI) by Dinh et al. follows a different approach than the previously discussed methods as it uses global statistics as an edge indicator [Dinh 11]. First, a feature space for edge detection is constructed. The new feature space has dimensionality n_D and in each component contains the spatial gradient magnitude measured at a pixel. On this feature space, ensemble clustering is performed. The combined use of a variation in clustering methods or parameters is expected to result in capturing a variety of structures in the data distribution that might be missed by a single clustering without prior knowledge.

The expected cluster size of a pixel is computed as the mean size of all clusters that contain said pixel and were obtained through different clusterings. This value is directly employed as a measure for edge strength, while directional information is not available. The rationale behind this concept is that edges are rare (or, as the authors put it, "salient") events in an image. Therefore a small expected cluster size is translated to a high probability of the pixel being located on an edge in the image.

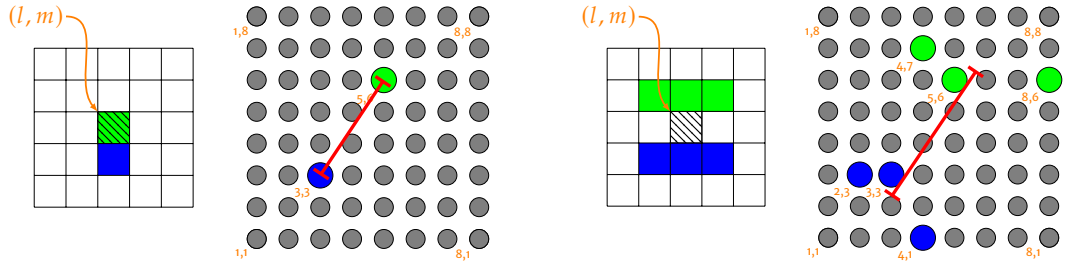
The ensemble clustering approach of SEDMI is problematic in terms of computational performance. The authors work around this problem by clustering only a small subset of the feature vectors and using a nearest-neighbor regression for estimating edge strength for the remaining pixels. The execution time of a Matlab implementations made available by the authors is still considerably high. Note that an alternative approach to this concept may be realized using a SOM. For each model vector, the amount of pixels represented by it can be easily computed. Then, following the conjecture of Dinh et al., this representation factor would translate to the edge strength.

4.1.2 Data-driven Pseudometric

The SOM is a useful tool for reducing the high dimensionality of a hyperspectral image in a data-driven fashion. As discussed in Chapter 3, It provides a low-dimensional topological representation of the contained spectra and their relationship. Instead of relying on the linearization of Toivanen et al., in our method we avoid the ordering of spectral vectors. Recall from Eq. 4.1 and Eq. 4.2 that an edge detection filter can be replicated using a high-dimensional dissimilarity measure. We can exploit the topological information of the SOM to produce such a measure. We gave an early example of such in Eq. 3.38. We will now define two new pseudometrics for spectra that are based on this concept. Furthermore, we formulate a 3×3 Sobel operator for both. Given a discrete two-dimensional image function $\mathrm{I}(l, m)$, Sobel is a linear filter for detecting edges in either horizontal or vertical direction,

$$\boldsymbol{S}_H = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad \boldsymbol{S}_V = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}, \tag{4.8}$$

which computes the first derivative along each axis, but includes a smoothing step before differentiation to reduce noise and obtain better continuity in the edge



(a) $\mathrm{SOM}_1\left(\boldsymbol{x}_{(l,m)}, \boldsymbol{x}_{(l,m+1)}\right)$ **(b)** $\mathrm{G_V}(l, m)$

Figure 4.4: Illustration of Sobel operator performed on SOM_1. In both (a) and (b), the image grid is shown on the left and the SOM is shown on the right.

input [Gonz 08, Chapter 10]. The combination of $\boldsymbol{S}_H$ and $\boldsymbol{S}_V$ then provides both edge magnitude and direction. This is input to the popular Canny edge detector which applies hysteresis thresholding for edge thinning [Cann 86].

Conventional BMU Lookup

The data-driven pseudometric SOM_1 is given as

$$\mathrm{SOM}_1(\boldsymbol{x}, \boldsymbol{y}) = \left\| \boldsymbol{r}^{(c(x))} - \boldsymbol{r}^{(c(y))} \right\|_2 , \tag{4.9}$$

which, in accordance to the neighborhood function (see Eq. 3.13), computes the Euclidean distance between the BMU locations of $\boldsymbol{x}$ and $\boldsymbol{y}$ in the SOM topology. It is easy to show that SOM_1 is a pseudometric, as the use of the L_2 norm on the SOM lattice fulfills the conditions in Eq. 2.6 (see page 21). However, Eq. 2.7 is not fulfilled as $\boldsymbol{x}$ and $\boldsymbol{y}$ both might share the same $\boldsymbol{m}_c$. In this case we expect any variation between the two pixels to originate from noise or numerical inaccuracies.

In the case of a 1-D map, this yields the same behavior as the method by Toivanen et al. [Toiv 03]. It differs for a 2-dimensional topology, where now the topological relationship of neurons is the sole origin of SOM_1. Besides avoiding artifacts introduced by a linearization, this approach bears two significant advantages that make it a generalization of the previous method. First, in the 2-D case we can lift the restrictions on SOM size that were imposed by the space filling curves. SOM size can now be arbitrary instead of being bound to a power of 4 or 9. Second, the SOM topology can be of higher dimensionality.

We now proceed to combine SOM_1 with the Canny edge detector, which takes the first derivatives of the image as input. Therefore, we create horizontal and vertical differential maps $\boldsymbol{G}_H$ and $\boldsymbol{G}_V$. Note that the method by Bakker and Schmidt (Eq. 4.2 on page 63) relies on the fact that there is only one negative component in the Laplace operator $\boldsymbol{L}$ (Eq. 4.1), which is not the case for the Sobel operators (Eq. 4.8). As dicussed, Sobel combines smoothing with differentiation. For the smoothing step, we may linearly combine locations on the SOM for both the negative and positive part of the filter separately. Then, the differentiation is carried out on the two compounds. Figure 4.4 illustrates this concept.

Each pixel $\boldsymbol{x}$ in the image is assigned to its match $\boldsymbol{m}_c$ (Eq. 3.9 on page 31). We then denote $\boldsymbol{r}^{(c(x))}$ for the pixel at l, m as $\boldsymbol{r}_{(l,m)}$. Four locations (left, right, top, bottom) based on the spatial neighborhood of each pixel are computed,

$$\begin{aligned}
\boldsymbol{r}^*_L &= \frac{1}{4}\left(\boldsymbol{r}_{(l-1,m-1)} + 2\boldsymbol{r}_{(l-1,m)} + \boldsymbol{r}_{(l-1,m+1)}\right) , \\
\boldsymbol{r}^*_R &= \frac{1}{4}\left(\boldsymbol{r}_{(l+1,m-1)} + 2\boldsymbol{r}_{(l+1,m)} + \boldsymbol{r}_{(l+1,m+1)}\right) , \\
\boldsymbol{r}^*_T &= \frac{1}{4}\left(\boldsymbol{r}_{(l-1,m-1)} + 2\boldsymbol{r}_{(l,m-1)} + \boldsymbol{r}_{(l+1,m-1)}\right) , \\
\boldsymbol{r}^*_B &= \frac{1}{4}\left(\boldsymbol{r}_{(l-1,m+1)} + 2\boldsymbol{r}_{(l,m+1)} + \boldsymbol{r}_{(l+1,m+1)}\right) .
\end{aligned} \tag{4.10}$$

Finally, the differential maps $\boldsymbol{G}_H$ and $\boldsymbol{G}_V$ are created via

$$\begin{aligned}
\mathrm{G_H}(l,m) &= \operatorname{sgn}\left(\|\boldsymbol{r}^*_L\|_2 - \|\boldsymbol{r}^*_R\|_2\right)\left\|\boldsymbol{r}^*_L - \boldsymbol{r}^*_R\right\|_2 , \\
\mathrm{G_V}(l,m) &= \operatorname{sgn}\left(\|\boldsymbol{r}^*_T\|_2 - \|\boldsymbol{r}^*_B\|_2\right)\left\|\boldsymbol{r}^*_T - \boldsymbol{r}^*_B\right\|_2 ,
\end{aligned} \tag{4.11}$$

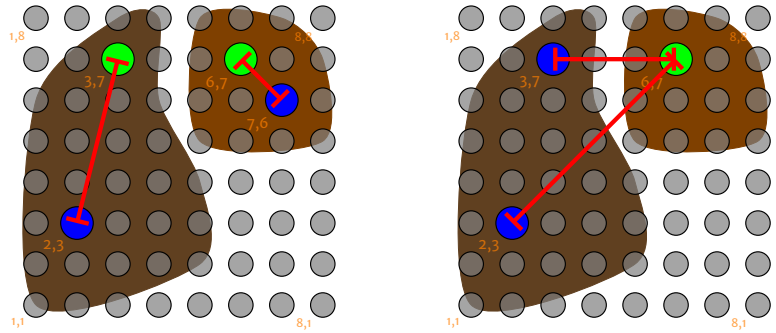
where $\operatorname{sgn}(\cdot)$ is the signum function. The described differential maps, in combination, can be used for both edge magnitude and direction.

This method works with a sharp query into the SOM. It was first published in [Jord 11]. In the experimental results we will see how the use of SOM_1 avoids most of the artifacts that are present in edge maps obtained by Toivanen's ordering. However, we find that in some cases, edge magnitudes appear imbalanced regarding the respective perceived discontinuities in the image. Analysis of this effect leads us to a second pseudometric, SOM_2, which is based on a broader query.

Multi-BMU Lookup

The distance of two nodes in the SOM is governed by two properties in the map: First, whether the nodes represent spectra from the same cluster (e. g. a material present in the scene) or two different clusters. As we learned in Chapter 3, clusters in the spectral distribution will find their designated connected areas in the map. This is the distinction that should result in an edge. Second, how big the representation of a cluster is in the map. Based on variety within a cluster (e. g. influence of noise in darker versus brighter areas of the image) and the amount of pixels that belong to each cluster, the amount of model vectors used by the SOM to represent a cluster varies. However, a bigger representation of a cluster should typically not result in an edge. Figure 4.5 illustrates how the influence of cluster size may be a problem for the edge detection through SOM_1. In the figure, a synthetic example for the SOM representation of two clusters is shown, whereas clusters are represented by 15 nodes, and 31 nodes, respectively. When comparing two spectra from these clusters, the distance of their respective BMUs in the map is affected by the cluster representation. For clarity, this illustration neglects the fact that a well-trained SOM exhibits smooth transitions between clusters.

While the calculation of SOM_1 relies on a single BMU of each spectral vector only, in Section 3.2.2 we discussed how we can obtain richer information from the SOM by extending the BMU lookup. Through the lookup of multiple BMUs, we now



(a) no edges **(b)** edges

Figure 4.5: Illustration of a SOM_1 weakness. Two clusters represented in the SOM are depicted by a brown background. In (a), we would expect a small distance for both comparisons. Likewise, in (b), we would expect a high distance for both comparisons.

define a pseudometric SOM_2 which is less affected by cluster size. To compute the dissimilarity for pixels $\boldsymbol{x}$, $\boldsymbol{y}$, we first obtain two sets of map locations $\mathcal{R}_x$, $\mathcal{R}_y$, where

$$\mathcal{R}_x = \mathcal{R}^{(\boldsymbol{c}^{(x)})} = \left\{ \boldsymbol{r}^{\left(c_j^{(x)}\right)} \,\middle|\, c_j^{(x)} \in \boldsymbol{c}^{(x)} \right\} , \tag{4.12}$$

and $\boldsymbol{c}^{(x)}$ is obtained by Eq. 3.15 (see page 37). The computation has a parameter n_C, which is the length of $\boldsymbol{c}$, or the number of BMUs to look up for each vector. For the similarity of $\mathcal{R}_x$ and $\mathcal{R}_y$, we use the so-called Earth Mover's Distance (EMD). In computer vision, EMD was first used by Rubner et al. to compute a perceptual distance between images by comparing their color signatures [Rubn 98]. Earlier works exist that use constraint variants of this distance [Ruzo 01]. The most prevalent application of EMD is histogram comparison. The EMD can be used to find a distance between two finite-sample distributions of same size, in our case coordinate sets. Intuitively, we compute the minimum amount of individual shifting needed on coordinates from one set to match the coordinates of the other set. The distance is posed as a minimization problem as follows. Coordinates from each set can be paired and then the sum of pair distances computed. The optimal pairing needs to be found w. r. t. the sum of pair distances. This problem is modeled in graph theory using a bi-partite graph with weighted edges. A modified Hungarian Method [Khul 99] finds a minimum-weight perfect matching in $\mathcal{O}\left(n_C^3 \log n_C\right)$. Faster approximations exist, bringing complexity down to linear time [Shir 08], however due to a low number of n_C typically employed, we do not further investigate them.

The pseudometric SOM_2 is then given as

$$\mathrm{SOM}_2(\boldsymbol{x}, \boldsymbol{y}) = \mathrm{EMD}(\mathcal{R}_x, \mathcal{R}_y) , \tag{4.13}$$

which, in the same vein as SOM_1, is a pseudometric as EMD is a metric.

Figure 4.6 illustrates how the distribution of additional BMUs in the respective originating clusters evens out the variance in cluster size. When the EMD between both groups is computed, we will see a significantly higher value for the edge cases

(a) no edges **(b)** edges

Figure 4.6: Illustration of SOM_2 behavior. Two clusters represented in the SOM are depicted by a brown background. Additionally to the first BMU, we see four secondary BMUs in a lighter shade. Secondary BMUs might overlap. In (a), we would expect a small distance for both comparisons. Likewise, in (b), we would expect a high distance for both comparisons.

than for the non-edge cases. In the latter case, parts of the BMUs from the two groups often overlap. We will show how it applies to real data in our experiments.

To combine SOM_2 with the Canny edge detector, we replicate the Sobel filter analogously to SOM_1. In this case, in the smoothing step, we can combine the BMUs of all three pixels corresponding to each side of the differentiation (illustrated in Figure 4.4). We query for $n'_C = \frac{1}{4}n_C$, and $n'_C = \frac{1}{2}n'_C$ for the outer pixels, and center pixel, respectively and obtain a multiset through concatenation. A multiset, or bag, allows multiple instances of its elements. The four multisets based on the neighborhood of each pixel are,

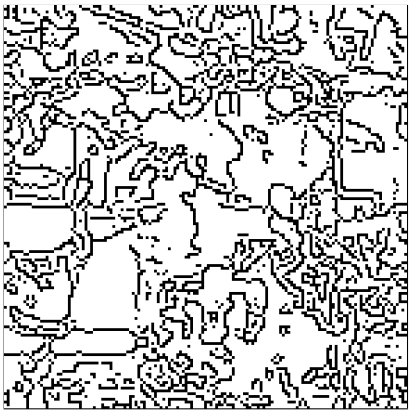
$$\mathcal{R}^*_L = \left\{ \begin{array}{l} \mathcal{R}_{(l-1,m-1),1}\,,\ldots,\mathcal{R}_{(l-1,m-1),n'_C}\,, \\ \mathcal{R}_{(l-1,m),1}\,,\ldots,\mathcal{R}_{(l-1,m),2n'_C}\,, \\ \mathcal{R}_{(l-1,m+1),1}\,,\ldots,\mathcal{R}_{(l-1,m+1),n'_C} \end{array} \right\}, \tag{4.14}$$

and $\mathcal{R}^*_R$, $\mathcal{R}^*_T$, $\mathcal{R}^*_B$ likewise. The differential maps $\boldsymbol{G}_H$ and $\boldsymbol{G}_V$ are created via

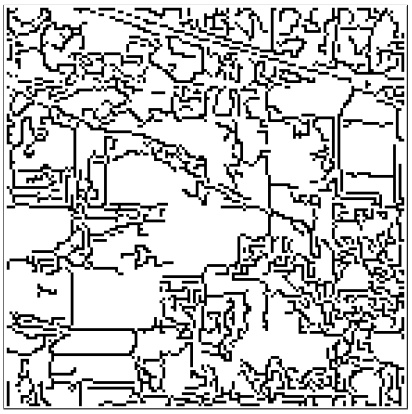
$$\begin{aligned} \mathrm{G_H}(l,m) &= \mathrm{sgn}\left(\|\boldsymbol{r}'_{(l-1,m)}\|_2 - \|\boldsymbol{r}'_{(l+1,m)}\|_2\right)\mathrm{EMD}(\mathcal{R}^*_L,\mathcal{R}^*_R)\,, \\ \mathrm{G_V}(l,m) &= \mathrm{sgn}\left(\|\boldsymbol{r}'_{(l,m-1)}\|_2 - \|\boldsymbol{r}'_{(l,m+1)}\|_2\right)\mathrm{EMD}(\mathcal{R}^*_T,\mathcal{R}^*_B)\,, \end{aligned} \tag{4.15}$$

(a) Image in true color

(b) SOM ordering, 64 × 1 SOM



(c) SOM ordering, 16 × 16 SOM



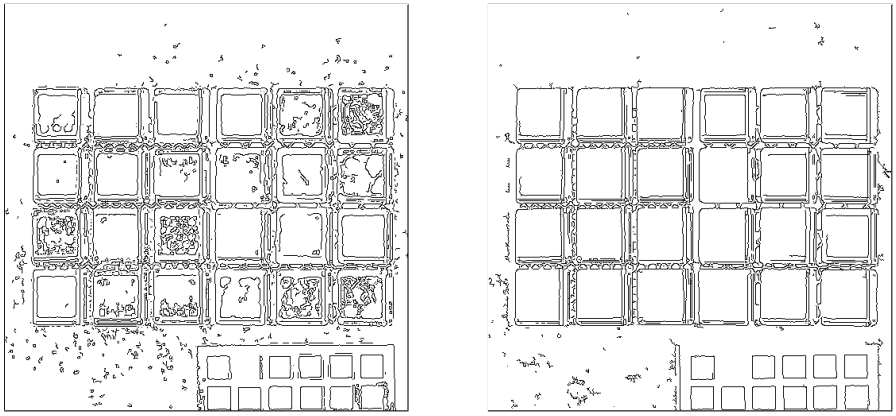
(d) SOM_1, 16 × 16 SOM

Figure 4.7: Canny edge detector results on *Indian Pines* image used for evaluation in [Toiv 03]. (b) is taken from the original publication, (c) was produced by our implementation of the SOM ordering, (d) was produced by the SOM_1 Sobel method using the same SOM training parameters.

where $\boldsymbol{r}'_{(l,m)}$ is the representative location of the pixel at l, m given by Eq. 3.16 and Eq. 3.19.

4.1.3 Experimental Results

In our evaluation of the proposed methods, we provide a qualitative comparison on multispectral and hyperspectral data against the state of the art. Unfortunately, we are not aware of any available ground-truth data for this task in this domain. However, in our comparison we can visually reveal strengths and weaknesses of each method on a range of images. When not stated otherwise, all edge detection results are printed such that white represents the lowest value (no edge) and black represents the highest value (strong edge) in each individual map. For better visibility, the 0.01 % highest values were masked out before defining the range for the intensity normalization.



(a) SOM ordering, 16 × 16 SOM **(b)** SOM_1, 16 × 16 SOM

Figure 4.8: Canny edge detector results on *Glass Tiles* image.

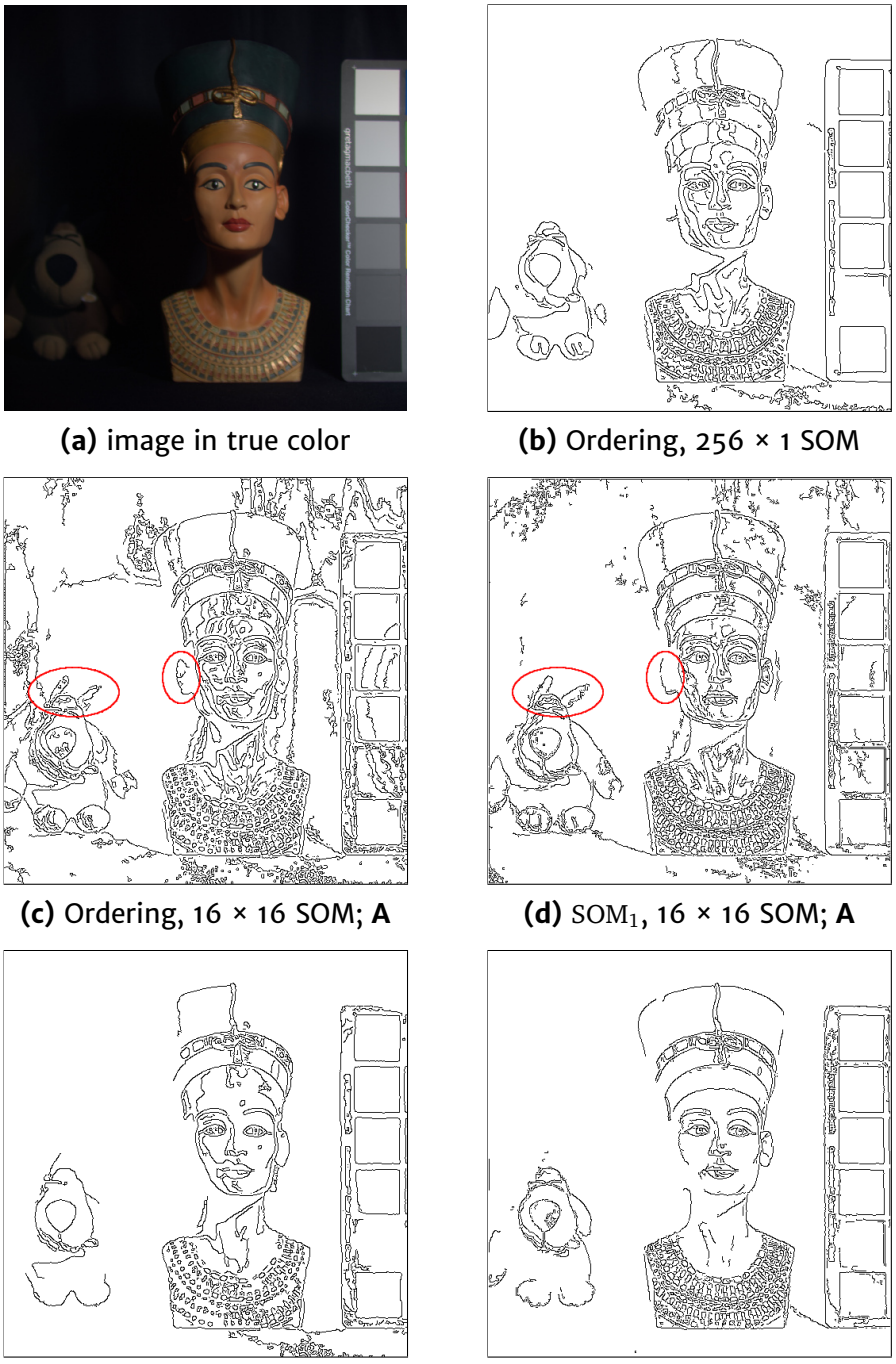
Methods based on the Self-organizing Map

We first compare SOM_1 with the method of Toivanen et al., denoted as *SOM ordering*, which is a precursor to our method in using a SOM for edge detection. Recall that the SOM ordering is first established on a 1-D or 2-D SOM before applying the Canny edge detector.

Figure 4.7 depicts a result published by Toivanen et al. on the *Indian Pines* remote sensing image next to our result using the SOM_1-based Sobel map. One can observe that SOM_1 helps to better discern the edges present in the image, most visible at the track on the top. Also, the rectangular structures are better reflected. Figure 4.8 shows the results of both methods on the multispectral image depicted in Figure 4.3a. Based on the flawed ordering seen in Figure 4.3b, strong edges are obtained between pixels of similar spectra. The edge detector cannot prevent false edges without losing valuable edge information. A problem that does not occur with SOM_1.

The *Egyptian Statue* image is an especially hard case for this task, as it contains structure that is either well illuminated or obscured in shadow areas of the scene. It depicts a bust of Nofretete next to a stuffed toy. We investigate the performance of Canny with both inputs from the SOM ordering and SOM_1 in Figure 4.9. Results are shown for two manually and individually selected Canny hysteresis threshold settings. As can be seen in the second row, the SOM manages to preserve structure that is in shadow and only perceptible in a small amount of spectral bands (marked in red). With the 1-D SOM, the contour of the shadowed throat is lost while smooth transitions in the face still lead to edges. A similar effect is visible with the 2-D SOM using a space filling curve. Contrary, the SOM_1 measure manages to expose characteristics of face, hat and toy while maintaining a low noise level. Find another result, on the *Fake and Real Food* image, in Appendix D on page 174.

Toivanen et al. found that on their two remote sensing images (one of which is shown in Figure 4.7), both 1-D SOM and 2-D SOM give comparable results [Toiv 03]. Our experiments reveal that when compared on a larger set of images, the 1-D SOM typically performs worse and the 2-D SOM ordering suffers from linearization artifacts. As a result, our proposed method achieves better edge detection on the tested images.



(a) image in true color

(b) Ordering, 256 × 1 SOM

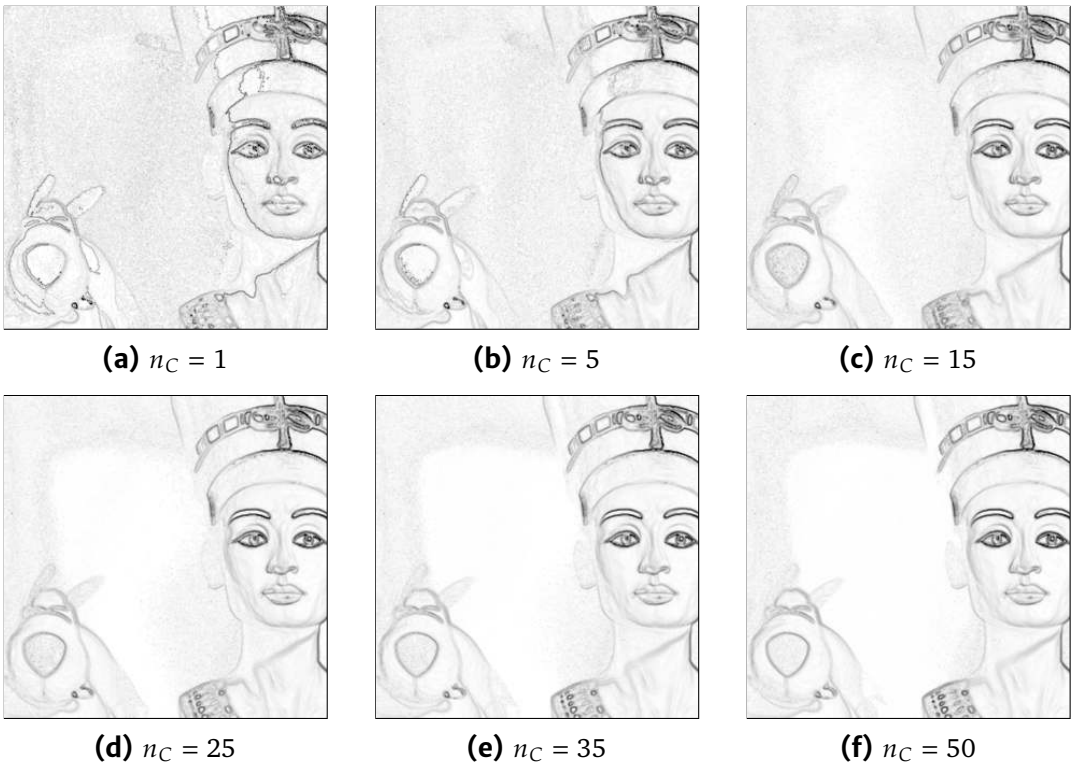
(c) Ordering, 16 × 16 SOM; **A**

(d) SOM_1, 16 × 16 SOM; **A**

(e) Ordering, 16 × 16 SOM; **B**

(f) SOM_1, 16 × 16 SOM; **B**

Figure 4.9: Canny edge detector results on *Egyptian Statue* image. Canny parameter criteria: **A**) best object contour preservation; **B**) minimum fine-grained noise introduced by object/background texture. Marked in red are object parts that are not visible in the true-color display. See Figure D.9 on page 178 for a helpful visualization of the image.



(a) $n_C = 1$ **(b)** $n_C = 5$ **(c)** $n_C = 15$

(d) $n_C = 25$ **(e)** $n_C = 35$ **(f)** $n_C = 50$

Figure 4.10: Laplace output for several choices of n_C on *Egyptian Statue* image. Results are shown on a cropped region for better visibility. $n_C = 1$ corresponds to SOM_1.

We proceed with an examination on how SOM_1 and SOM_2 relate performance-wise. Our expectation is that the latter pseudometric leads to a well-balanced edge map where the former suffers from the effect of different cluster representations in the map. For the desired effect, we have to determine a generally suitable number of BMUs to lookup n_C. We use the Laplace edge filter (see Eq. 4.2), setting $\mathrm{d} = \text{SOM}_2$ with different choices for n_C. Note that the case $n_C = 1$ directly corresponds to SOM_1. While in the previous tests, we operated on a SOM of size $n_M = 256$ for better comparability with the method by Toivanen et al., we now train a 2-D SOM of our preferred size of $n_M = 1024$ model vectors.

Figure 4.10 depicts results on a detail of the *Egyptian Statue* image from Figure 4.9a. Note the difficulty of the task, given the bad illumination in the scene for most parts of the detail. We observe that while the result for $n_C = 1$ is already quite decent, it still includes some artifact edges and the edge magnitudes sometimes appear not well-balanced, which includes exposing noise in the background. When increasing the number of looked up BMUs, we see three effects. First, the artifact edges disappear. Second, edges concentrate more on material separation than illumination effects, visible in the left side of the statue face. Third, we see an overall reduction in noise. The edge maps stabilize for higher numbers of n_C. Some true edges slowly start to fade out. This is expected, as a high count of BMUs inevitably leads to non-relevant BMUs being chosen when a cluster is represented by less BMUs, effectively adding a random baseline for both pixels in each measurement.

As a result of this comparison, we fix $n_C = 25$ for all further testing. Note that we can expect a good choice of n_C to be independent of the input data, but rather only affected by the size of the SOM.

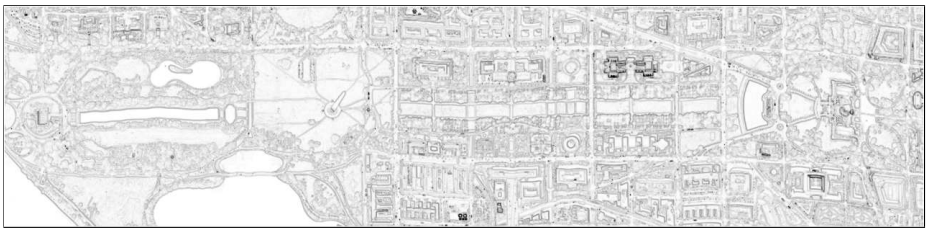
The results presented so far indicate that by utilizing a SOM, we can obtain a high-quality data-driven dissimilarity measure that provides sensible input for an edge detector. Furthermore, we will now show how our measure competes against established, heuristic measures in edge detection on a wider range of images.

Hyperspectral similarity measures

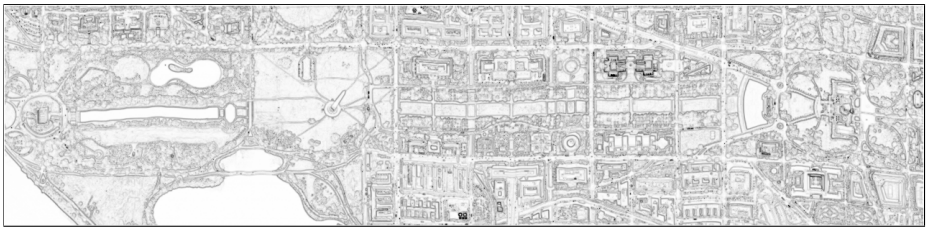
The Laplace filter is a good instrument for evaluating the behavior of SOM_1 and SOM_2 in comparison to established hyperspectral similarity measures, namely L_2, SA, SID, and the measure introduced for RCMG, SAL_∞.

Figure 4.11 shows edge maps computed on the *D.C. Mall* remote sensing image. While the general edge response is satisfactory, a weakness of L_2, SA, SAL_∞, and SID on this image is the overly strong response on shiny roofs present in the scene, which diminishes other edges in the image that could be considered equally important. SID is worst in this regard. The new measures SOM_1 and SOM_2 provide the most balanced output for this image. SOM_1 shows noise artifacts, visible in the water area on the left bottom of the image, which are not present with SOM_2.

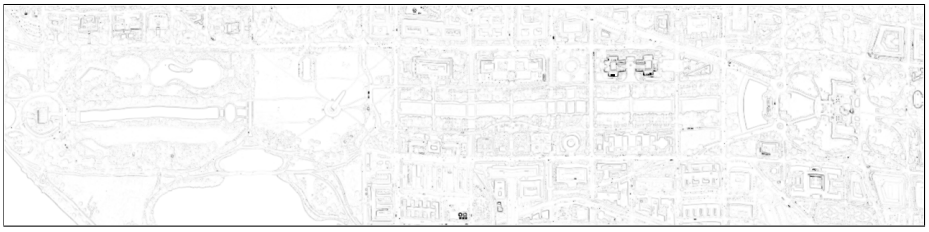
Figure 4.12 shows edge maps computed on the *d3* image from the *Harvard* dataset, depicted in Figure 2.6d on page 15. We chose this image as it depicts a scene in a natural environment and contains objects of different materials and shape stacked on each other, resulting in a variety of edge detection challenges. In the result for L_2 we see the strong effect of mean intensity on the Euclidean distance. Specular highlights induce strong edge responses, e. g. on the red plastic bottle on the bottom shelf (marked in red). In contrast, the black cables in front of gray book pages on the middle shelf (also marked in red) elicit a very weak edge response. Other than one might expect, SA performs very poorly for edge detection on this image. The measure mostly covers the noise in the image, which is more dominant on the vector angles in darker regions, leading to dark regions being misinterpreted as edges, while real edges in the scene give almost no response. The combined measure SAL_∞ adds edges that are also present in L_2, but generally also fails to clearly separate discontinuities in the scene from homogeneous (when disregarding the noise) areas. SID has the property that zero vectors cannot be compared, which results in a small amount of white pixels spread over the image. Apart from this defect, which may affect some algorithms more than others, SID is also very prone to the noise in darker regions. Even more so than with SA and SAL_∞, we can recognize a vertical stripe pattern in sensor noise on the SID output, most visible in the area marked in red. Finally, SOM_1 and SOM_2 both provide reasonable edge maps on this image. SOM_2 beats SOM_1 in certain areas of the image, visible e. g. on the semi-transparent plastic bottle on the lower shelf (as referenced before when discussing L_2) and is less prone to noise in dark, homogeneous areas.



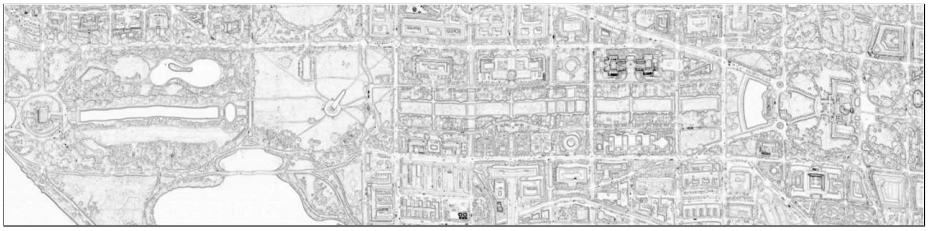
(a) L_2



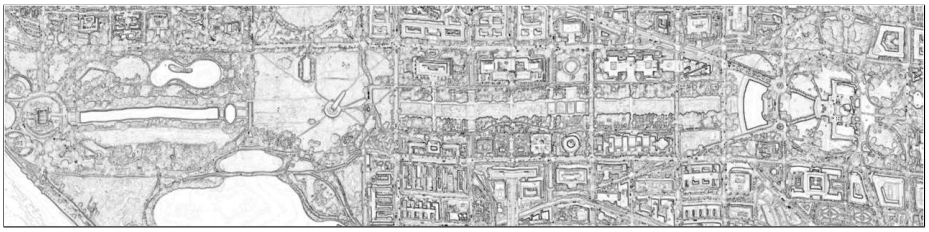
(b) SA



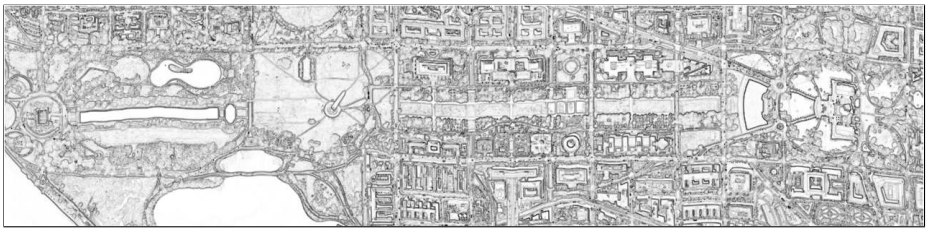
(c) SID



(d) SAL_∞



(e) SOM_1



(f) SOM_2

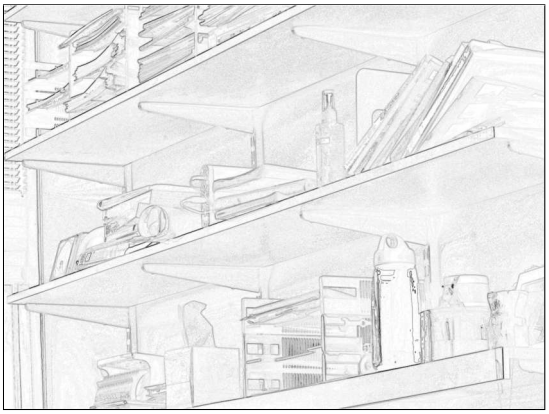
Figure 4.11: Laplace edge detector results on *D.C. Mall* image (see Fig. 2.6e, page 15).

(a) L_2

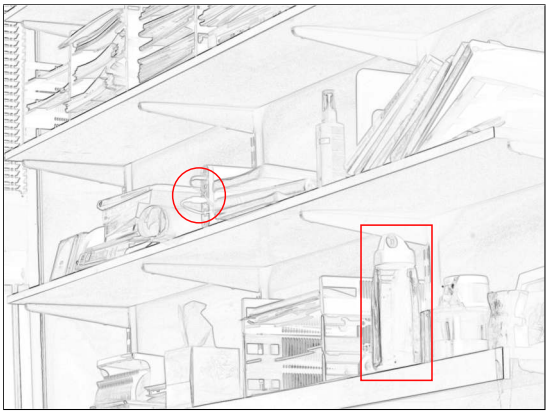
(b) SA

(c) SID

(d) SAL_∞

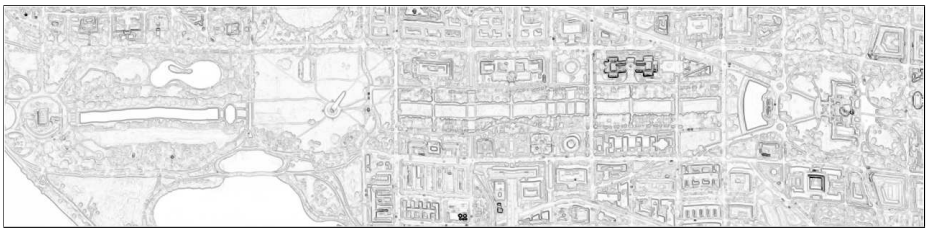


(e) SOM_1

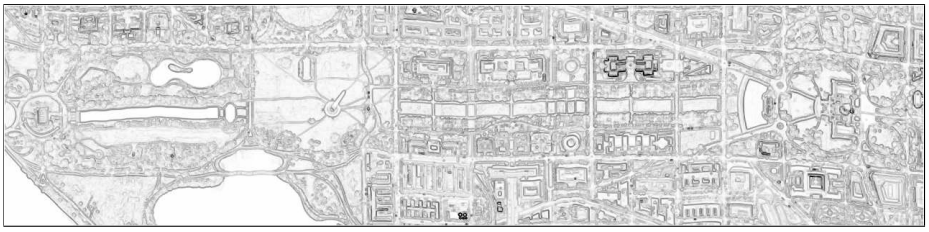


(f) SOM_2

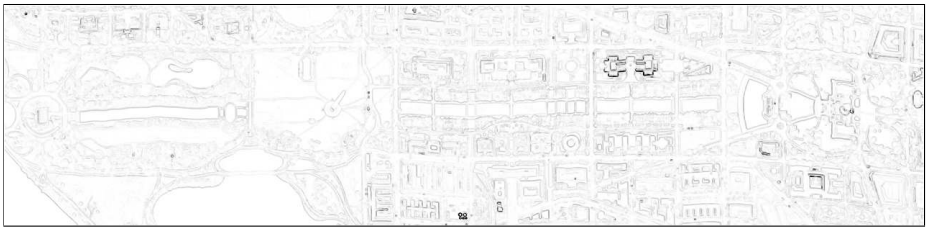
Figure 4.12: Laplace edge detector results on *d3* image from the Harvard dataset (see Figure 2.6d, page 15). Red markers are referred to in the text.



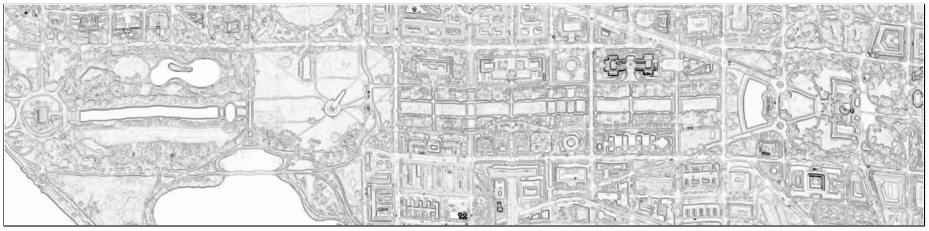
(a) L_2



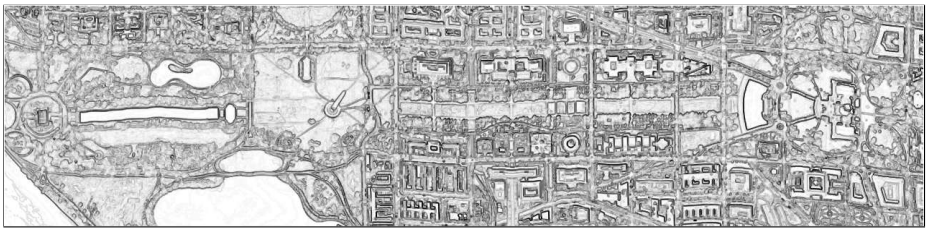
(b) SA



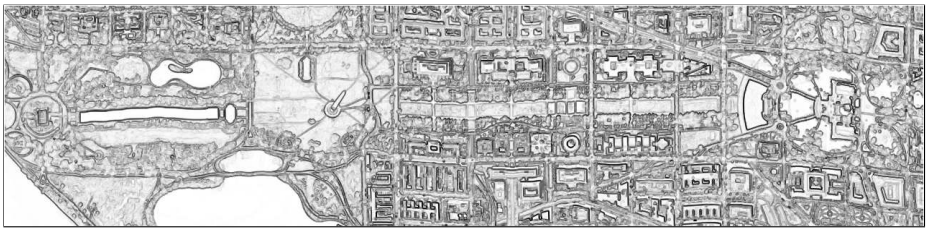
(c) SID



(d) SAL_∞

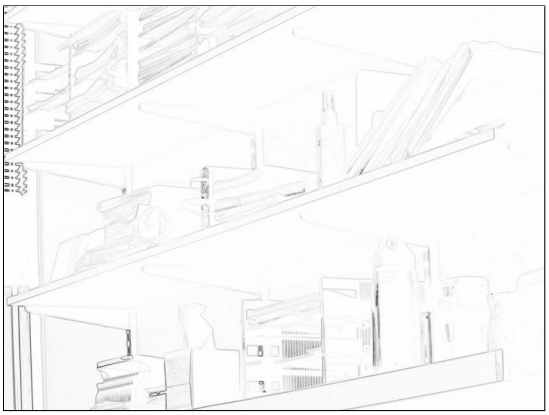


(e) SOM_1



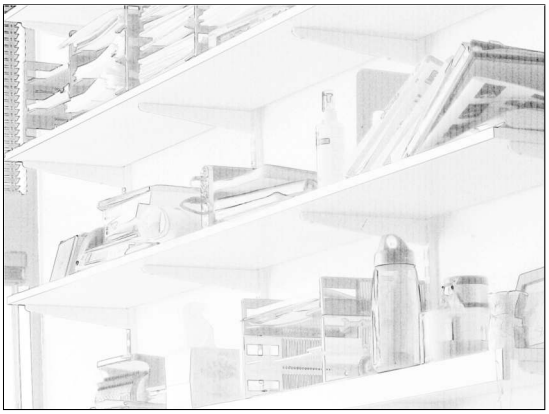
(f) SOM_2

Figure 4.13: RCMG edge detector results on *D.C. Mall* image (see Fig. 2.6e, page 15).



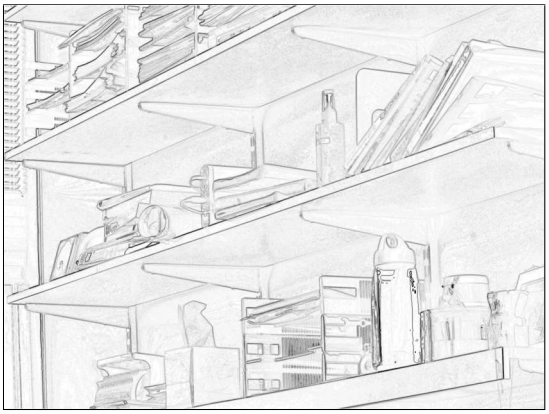
(a) L_2

(b) SA

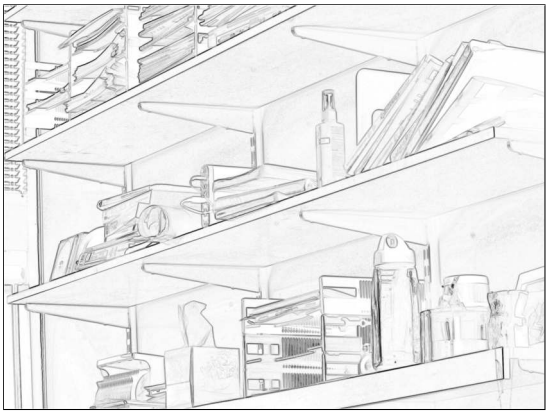


(c) SID

(d) SAL_∞



(e) SOM_1



(f) SOM_2

Figure 4.14: RCMG results on *d3* image from the Harvard dataset (see Figure 2.6d, page 15).

Robust Color Morphological Gradient

In this test, it is determined whether the hyperspectral extension of RCMG can deliver the same high performance that RCMG is know for on color images [Mitt 12]. Note that we do not evaluate RCMG against our method, but rather combine them: Both SOM_1 and SOM_2 can be efficiently used as input to the RCMG algorithm. We run RCMG on them alongside L_2, SA, SID, and SAL_∞, as in the previous test.

Figure 4.13 shows RCMG edge maps computed on the *D.C. Mall* remote sensing image. We see that the results do not differ much from the Laplace results for this image. Edges appear more pronounced in the RCMG images. This also slightly amplifies the noise in the SOM_1 output we had discussed before.

Figure 4.14 shows RCMG edge maps computed on the *d3* image. When comparing with the Laplace edge filter results in Figure 4.12, we make several observations. The outlier removal manages to remove problematic values from the SID input. However, as with SA and SAL_∞, noisy regions overshadow the response at discontinuities in the scene. In SAL_∞, the edge output does improve when compared to the Laplace result, however edges that are clearly missing in Figure 4.14a are also missing in Figure 4.14d. The best result is achieved by SOM_2, followed by SOM_1. We observe that RCMG enhances the overall contrast of edges when compared to the Laplace filter applied on the same measures.

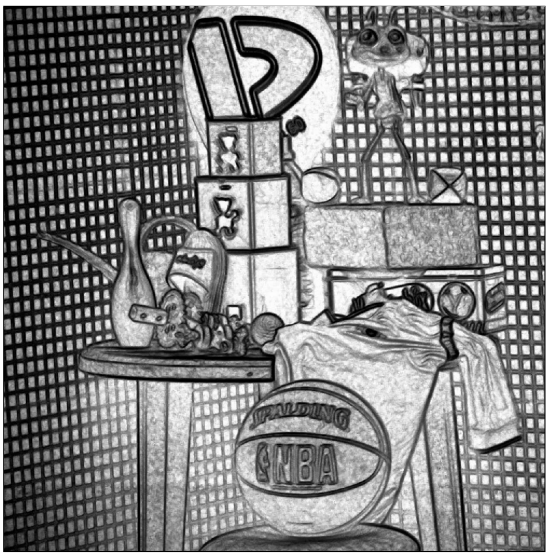
While RCMG can provide a greatly enhanced gradient map, it is not able to overcome the inherent weaknesses of its input. In the case of L_2, we miss low-intensity details, or changes that are only found in a few bands. In the case of SA, noise in low-intensity regions is greatly amplified and leads to false edges. The output of RCMG is determined by the input metric: it is the maximum pairwise distance in a subset of neighboring pixels. Therefore, the distances are neither amplified for regions where the overall intensity output is weak, nor reduced in regions that are overall noisy. We see that computing the pairwise distances with our learning-based pseudometric significantly improves the RCMG performance.

SEDMI and Binary Edge Maps

SEDMI is interesting for comparison as it is a recent edge detection method designed specifically for hyperspectral images. For SEDMI, results are published for visual comparison on the *Scene 5* lab image by Foster [Dinh 11]. Figure 4.15 depicts the result reported by Dinh et al. alongside our RCMG results on L_2 and SOM_2. Similar to our previous observations, other measures did not prove beneficial as input to RCMG on this image. We see that for this image, RCMG shows a weak response on some important edges, but retains others. The SEDMI output shows particularly thick edges, which equals a loss of resolution. Similarly thick edges can be produced with RCMG using a larger structuring element. In general, apart from computational complexity, SEDMI provides competitive results. A drawback of the method seems to be the noise present on many of the homogeneous surfaces in the image. In particular, we notice the face of the toy figure to the top right.

As a final experiment, we use the Sobel operator defined in Eq. 4.15 as input to the Canny edge detector, which provides binary edge maps. As the SOM_2 Sobel

(a) image in true color



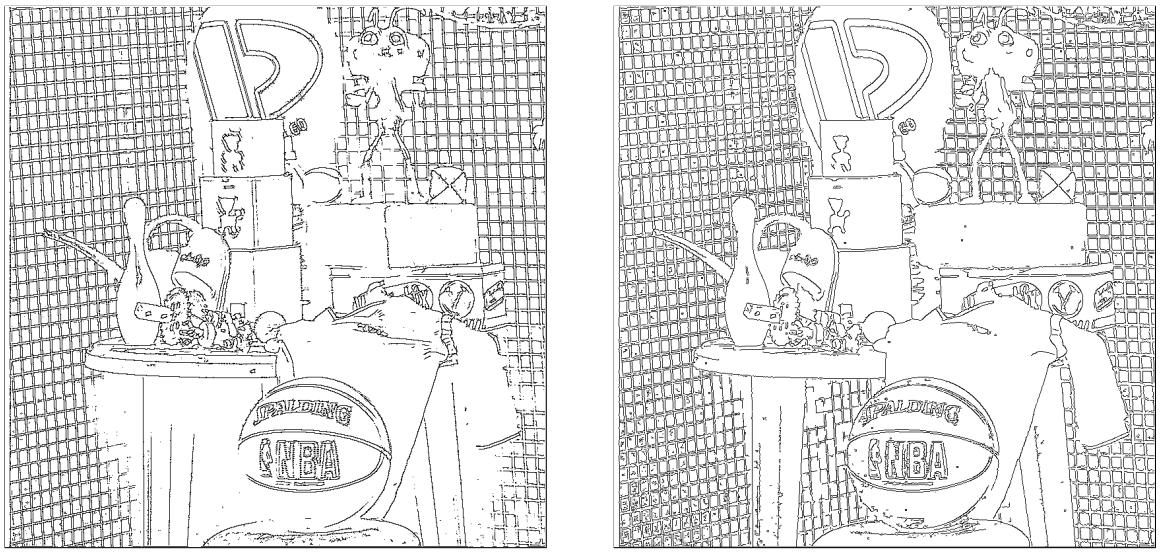
(b) SEDMI [Dinh 11]

(c) RCMG on L_2

(d) RCMG on SOM_2

Figure 4.15: SEDMI and RCMG results on the *Foster* lab image.

operator allows only multiples of four in the choice of n_C, we set $n_C = 24$ instead of $n_C = 25$ as in the previous tests. In Figure 4.16 we compare the result with the one reported by Dinh et al., where the authors created a binary map with an edge thinning method and hand-picked a threshold parameter [Dinh 11]. Likewise, we hand-pick the hysteresis parameters of Canny to find a good compromise between missed edges and noise in the edge map. We observe that SOM_2, as expected, retains more details in dark regions of the image. For example, the basketball is clearly separated against the background in Figure 4.16b, which is not the case in Figure 4.16a, but clearly visible in Figure 4.15a. Instead, SEDMI shows some edges within the same object, introduced by geometry, that are not present in the SOM_2 output. As SEDMI does not provide edge direction for the edge thinning operation, its binarization is also prone to double edge artifacts.



(a) SEDMI [Dinh 11] **(b)** SOM_2

Figure 4.16: Binary edge detection results on the *Foster* lab image.

It has been shown that existing edge detection algorithms, based on the well-known metrics and dissimilarity measures often applied on hyperspectral data, fail to provide a consistent performance over a range of images from different settings. Our measures outperform them based on the fact that they are data-driven: rather than on the high-dimensional feature space as a whole, edges are found on the manifold that contains the spectra observed in the image, learned through the SOM. More results can be found in Appendix D on page 175.

4.2 Clustering

While edge detection can provide a spatial understanding of a captured scene, image segmentation allows to not only discern spectra at boundary regions, but also to identify regions of interest in the scene. Clustering is a means to segmentation that provides a structured view of the data by identifying clusters in the high-dimensional distribution and linking observed spectra to their respective cluster (hard assignment) or several clusters (soft assignment). Next to the distance or similarity of spectra, spatial relationships of pixels can also be of use in identifying clusters.

In the hyperspectral domain, clustering can help with finding material prototypes contained in the scene, or with identifying specific reflectance effects. For example, surfaces can be compactly clustered by their reflectance behavior if capturing parameters and data descriptors are appropriately chosen. Thus, areas of diffuse reflection, specular highlights, and shadows each form individual clusters of feature vectors. The distance between these clusters (and thus their distinguishability) may vary across surface materials. In general, surface materials are better discerned in the case of diffuse reflection than in the case of specular highlights or shadows.

In this section we discuss methods to find a partitioning of a multispectral or hyperspectral image in an unsupervised manner. In general, finding the appropriate clusters in the high-dimensional space without prior knowledge can be challenging. We will put most emphasis on the *mean shift* algorithm and its variants [Coma 02], as it is one of the more flexible clustering approaches. We can obtain high-quality segmentation results in several multispectral scenarios. However, mean shift is computationally demanding for higher spatial and spectral resolutions. We tackle this problem by finding means to reduce complexity without reducing the dimensionality of the input data. We also take a particular look at methods to exploit the self-organizing map for clustering and propose a new algorithm for clustering based on the topological information contained in the SOM.

4.2.1 Related Work

Two general criteria that form the basis of most algorithms are the maximization of intra-cluster similarity and minimization of inter-cluster similarity. Several directions exist to find a segmentation of the image that best fulfills these criteria, of which we mention the three most prominent ones.

Partitioning algorithms find a partition of the image, typically iteratively, based on an objective function. Most prominent is the *k-means* algorithm that forms partitions over a fixed number of centroids [Lloy 82]. Model-based partitioning maximizes the likelihood of the samples using a parametrized model, e. g. a mixture of multi-variate Gaussians, which can be solved using the expectation-maximization algorithm. For these algorithms to succeed, the data needs to fit the model, which in the case of k-means is hyperspherical clusters.

Hierarchical algorithms decompose the problem in a bottom-up (agglomerative) or top-down (divisive) fashion. In most cases, a level in the hierarchy needs to be chosen to obtain a final segmentation result. Prominent in hierarchical clustering are methods based on graph theory. The method by Felzenszwalb and Huttenlocher is agglomerative, i. e. it starts with a cluster for each sample [Felz 04]. In each level of the hierarchy, clusters are merged based on aforementioned intra-class and inter-class similarity criteria. An example of a divisive algorithm, also graph-based, is Normalized Cuts by Shi and Malik [Shi 97]. They find a bipartition of the graph via eigenvalue calculation. Then, the two obtained segments are recursively split via a stability criterion. Both methods rely on edge weights based on pairwise similarity. A divisive hierarchical algorithm based on optimization is found in deterministic annealing [Rose 98]. Huynh and Robles-Kelly apply deterministic annealing to multispectral image clustering, namely material clustering [Huyn 10b]. The algorithm is supposed to converge faster than its precursor, simulated annealing, and be less sensitive to initialization as k-means. It minimizes a cost function for finding (material) prototypes for the data, whereas the annealing temperature T weighs the level of randomness of pixel-prototype association. Initially, T is high and a single prototype is optimized. When T is lowered, the system splits prototypes, whereas at a constant temperature, the algorithm converges to a thermal equilibrium. The algorithm can be terminated when the desired amount of prototypes were found and converged.

Density-based algorithms consider regions of high density in the feature space as clusters. The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm finds clusters through the connectivity of samples [Este 96]. For this, the density around each point is defined by the number of samples that fall inside a hypersphere centered at the point. The point is considered an interior point if a minimum number of samples are located in the hypersphere. Interior points that are linked through a chain of overlapping hyperspheres form a cluster. Unfortunately, DBSCAN is not well suited for high-dimensional data [Xu 05]. The *mean shift* algorithm is conceptually based on kernel density estimation [Coma 02]. Starting at each sample point, it iteratively seeks for the next mode in the sample distribution. Then, clusters are formed by all points that share a mode. It has been shown to be effective in high dimensions [Geor 03]. A general advantage of density-based methods is that they neither rely on a specific data model, nor on spatial relationships of pixels.

We will now go into more detail on three algorithms for scene segmentation, representing these three directions, including superpixel segmentation (agglomerative hierarchical), k-means (partitioning), and mode seeking (density-based), before discussing existing SOM-based clustering algorithms.

Superpixels

Superpixel is a term recently coined in the field of computer vision. It describes a set of pixels that are spatially connected and share high similarity, replacing the rigid structure of the pixel grid of an image. A wide range of segmentation methods fall into the superpixel category, most prominently several graph-based algorithms as well as gradient-ascend methods. Achanta et al. [Acha 12] discuss and compare a comprehensive selection of these algorithms.

The method by Felzenszwalb and Huttenlocher (FH04) is a good representative example of this algorithmic family [Felz 04]. The algorithm operates on a 4-connected or 8-connected graph that represents each pixel $\boldsymbol{x}_i$ as a node $v_i \in \mathcal{V}$. Edges e_{ij} are weighted based on pixel dissimilarity, $w_{ij} = \mathrm{d}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The graph is then partitioned starting with one partition, or superpixel, per node. Superpixels are iteratively merged such that the final segmentation fulfills two properties: Edges between two vertices in the same component should have relatively low weights, while edges between to vertices in different components should have higher weights.

For this, the internal difference $\delta_{\mathrm{Int}}(\mathcal{L})$ of each component $\mathcal{L} \subseteq \mathcal{V}$ is defined as the largest weight in the minimum spanning tree (MST) of the component. The MST is a subset of all edges between vertices in $\mathcal{L}$ such that no edge can be removed without disconnecting one or more vertices from the others, and no edge can be replaced by another edge without increasing the sum of edge weights. The joint minimum internal difference of two components is defined as

$$\delta_{\mathrm{MInt}}(\mathcal{L}_1, \mathcal{L}_2) = \min\Big(\delta_{\mathrm{Int}}(\mathcal{L}_1) + \tau(\mathcal{L}_1), \delta_{\mathrm{Int}}(\mathcal{L}_2) + \tau(\mathcal{L}_2)\Big)\,, \tag{4.16}$$

where the threshold function τ is discussed below.

The difference between two adjacent components $\delta_{\mathrm{Dif}}(\mathcal{L}_1, \mathcal{L}_2)$ is the minimum weight w_{ij} of any edge connecting two vertices $v_i \in \mathcal{L}_1$ and $v_j \in \mathcal{L}_2$. This measure

might be seen as limited, only taking the smallest edge weight between the components into account. It was chosen for the sake of deriving a computationally efficient algorithm. The pairwise comparison predicate $\mathrm{D}(\mathcal{L}_1, \mathcal{L}_2)$ then decides whether two adjacent components are merged,

$$\mathrm{D}(\mathcal{L}_1, \mathcal{L}_2) = \left[\delta_{\mathrm{MInt}}(\mathcal{L}_1, \mathcal{L}_2) > \delta_{\mathrm{Dif}}(\mathcal{L}_1, \mathcal{L}_2)\right] . \tag{4.17}$$

In other words, two adjacent components are merged if any of their internal differences is larger than the difference between the components. If however, their internal differences are smaller, a boundary between the components is assumed. However, internal differences are offset by the threshold function

$$\tau(\mathcal{L}) = \frac{c}{|\mathcal{L}|} , \tag{4.18}$$

where c is a constant parameter, whereas a larger c causes a preference for larger components. The threshold function also tells us that stronger evidence for a boundary is required for small components, i. e. merging is favored for smaller components. Another parameter of the algorithm is m, the enforced minimum size of $\mathcal{L}$. For $m = 1$, arbitrarily small components may be included in the final segmentation, if they differ considerably from their neighboring components.

As a shared property between all superpixel segmentations, this technique is based on the spatial layout of the scene. In our application, which is typically clustering for material or specific reflectance effects, we strive for independence from the spatial layout to some degree. Two objects of same material in positions remote to each other in the scene are expected to fall into the same segment. Superpixels do not fit this premise. We later introduce means to exploit the method by Felzenszwalb and Huttenlocher for such a scenario in Section 4.2.2 and Section 4.2.3.

k-Means

An efficient clustering algorithm that does not take spatial relationships into account is *k-means*, a term that is simultaneously used for the problem to solve and the prominent algorithm that solves it. The k-means problem is to find n_K disjoint sets of input samples, whereas the mean of each set is optimal in terms of its Euclidean distance to all samples in the set, serving as its prototype. The most prominent algorithm solving this problem was proposed by Lloyd [Lloy 82]. It provides a local search solution to the problem that is known for its fast convergence and is a simple, hard-assigment variant of Expectation Maximization clustering [Bish 06, Chapter 9].

Several methods exist to initialize the set of cluster centers $\mathcal{M}, |\mathcal{M}| = n_K$ based on the input samples $\boldsymbol{x}_i$. A very basic approach is to randomly pick n_K input samples. Arthur and Vassilvitskii offer a more sophisticated initialization, coining their algorithm k-means++ [Arth 07]. They also select cluster centers randomly, however with a probability function that takes the distance of sample points to previously selected cluster centers into account. After initialization, the following steps are carried out iteratively.

In the E-step, each sample is assigned to one cluster center by setting the expectation of weight w_{ji} to

$$E[w_{ji}] = \left[j = \min_k \|\boldsymbol{\mu}_k - \boldsymbol{x}_i\|_2 \right] , \tag{4.19}$$

where $[\cdot]$ is the Iverson bracket. Then, in the M-step, cluster centers are re-computed as the average of all samples expected to be part of each cluster,

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n_X} E[w_{ji}]\boldsymbol{x}_i}{\sum_{i=1}^{n_X} E[w_{ji}]} . \tag{4.20}$$

These steps are repeated until convergence. Then, we can assign the label l_i to each sample as

$$l_i = \operatorname*{argmin}_j \|\boldsymbol{\mu}_j - \boldsymbol{x}_i\|_2 . \tag{4.21}$$

The computational complexity of the algorithm is a moderate $O(n_K n_X)$. Typically, a rather low n_K is chosen and only a small number of iterations is needed for convergence. Therefore, k-means is fast to compute. With larger n_K, e. g. for obtaining an under-segmentation as a pre-processing step, the speed advantage of k-means diminishes. The solution found is highly dependent on the initialization of cluster centers [Arth 07]. A common workaround is to re-run the algorithm several times with different initializations. Then, the compactness measure is used to select the final solution,

$$\mathrm{C}(\mathcal{X}, \mathcal{M}) = \sum_{i=1}^{n_X} \|\boldsymbol{x}_i - \boldsymbol{\mu}_{l_i}\|_2 , \tag{4.22}$$

which is the objective function solved by k-means. The compactness can be used as a general measure for clustering quality [Vesa 00a], however it is biased. The measure (and therefore k-means) is based on the assumption of spherically shaped clusters with a common diameter, e. g. isotropic Gaussian-distributed clusters. This is often not true for remote sensing images.

The number of clusters needs to be known beforehand. This makes n_K important prior knowledge that may not be available. In remote sensing, the similar *ISODATA* algorithm finds use, which incorporates splitting and merging of clusters [Lill 14, Chapter 7]. These operations are triggered by simple thresholds on the amount of data points associated to a cluster, the distance between cluster centers, and the variance within a cluster [Ball 65]. The thresholds are parameters which also need to be carefully chosen.

Mean Shift

The *mean shift* algorithm is a well-understood and popular clustering method that is in theory applicable to high-dimensional data, such as multispectral pixel vectors [Coma 02]. It was first presented in 1975 by Fukunaga and Hostetler [Fuku 75]. Mean shift is a density gradient estimator. It finds the modes of the multivariate distribution underlying a feature space with a kernel estimator. In our case, the

feature space consists of the n_X spectral vectors $\boldsymbol{x}_i$ of length n_D. The clustering is obtained by associating each pixel with a mode.

Mean shift is based on the concept of Kernel Density Estimation (KDE) [Coma 02]. KDE estimates the probability density function as a mixture of the observed samples $\boldsymbol{x}_i$. For each point $\boldsymbol{x}$ in $\mathbb{R}^{n_D}$, the Parzen-window estimate is given as

$$\hat{f}(\boldsymbol{x}) = \frac{1}{n_X} \sum_{i=1}^{n_X} \frac{1}{h_i^{n_D}} \kappa\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h_i}\right) , \tag{4.23}$$

where the window function $\kappa(\cdot)$ is a kernel. It has compact support and satisfies $\int_{\mathbb{R}^d} \kappa(\boldsymbol{x}) = 1$, $\kappa(\boldsymbol{x}) \geq 0$, and $\kappa(\boldsymbol{x}) = \kappa(-\boldsymbol{x})$. We can now restrict ourselves to a radially symmetric kernel with a 1-D *profile* $k(\cdot)$: $\kappa(\boldsymbol{x}) = c_k k(\langle \boldsymbol{x}, \boldsymbol{x} \rangle)$, whereas c_k is a normalization factor. Without loss of generality, we can investigate the case of $h_i = h$. By employing the profile notation, the KDE can then be formulated as

$$\hat{f}_{h,\kappa}(\boldsymbol{x}) = \frac{c_k}{n_X h^{n_D}} \sum_{i=1}^{n_X} k\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|_2\right) . \tag{4.24}$$

We continue with the density gradient estimate,

$$\hat{\nabla} f_{h,\kappa}(\boldsymbol{x}) = \frac{2c_k}{n_X h^{n_D+2}} \sum_{i=1}^{n_X} (\boldsymbol{x} - \boldsymbol{x}_i) k'\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|_2\right) , \tag{4.25}$$

which forms the basis of the mean shift filter, as explained below. We introduce the kernel $\gamma(\boldsymbol{x}) = c_g g(\langle \boldsymbol{x}, \boldsymbol{x} \rangle)$. The profile of γ, $g(\cdot)$, is defined as $g(\boldsymbol{x}) = -k'(\boldsymbol{x})$. The density gradient estimate using γ is

$$\begin{aligned} \hat{\nabla} f_{h,\kappa}(\boldsymbol{x}) =& \frac{2c_k}{n_X h^{n_D+2}} \sum_{i=1}^{n_X} (\boldsymbol{x}_i - \boldsymbol{x}) g\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|_2\right) \\ =& \underbrace{\frac{2c_k}{n_X h^{n_D+2}} \left(\sum_{i=1}^{n_X} g\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|_2\right)\right)}_{1} \underbrace{\left(\frac{\sum_{i=1}^{n_X} \boldsymbol{x}_i g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|_2\right)}{\sum_{i=1}^{n_X} g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|_2\right)} - \boldsymbol{x}\right)}_{2} . \end{aligned} \tag{4.26}$$

We observe that the first term of Eq. 4.26 is proportional to $\hat{f}_{h,\gamma}(\boldsymbol{x})$. The second term is the *mean shift* vector,

$$\boldsymbol{m}_{h,\gamma}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n_X} \boldsymbol{x}_i g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|_2\right)}{\sum_{i=1}^{n_X} g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|_2\right)} - \boldsymbol{x} = \frac{1}{2} h^2 c \frac{\hat{\nabla} f_{h,\kappa}(\boldsymbol{x})}{\hat{f}_{h,\gamma}(\boldsymbol{x})} , \tag{4.27}$$

where $c = \frac{c_g}{c_k}$. It corresponds to a shift to the center of mass in the current window, weighted by $g(\cdot)$. This shift is performed iteratively until convergence, i. e. $\boldsymbol{m}_{h,\gamma}\left(\boldsymbol{x}^{(s)}\right) = \boldsymbol{x}^{(s+1)} - \boldsymbol{x}^{(s)}$ for each iteration s. Mean shift converges at a stationary point of $\hat{f}_{h,\kappa}$,

$$\boldsymbol{m}_{h,\gamma}\left(\boldsymbol{x}^{(s)}\right) = 0 \Rightarrow \hat{\nabla} f_{h,\kappa}\left(\boldsymbol{x}^{(s)}\right) = 0 , \tag{4.28}$$

due to Eq. 4.27. To conclude, starting at $\boldsymbol{x}^{(1)}$, we perform a gradient ascend that converges at a mode of the distribution. Two choices are popular for the kernel κ with corresponding γ. κ is called the *shadow* of γ [Coma 02]. One, the Epanechnikov kernel,

$$k(t) = \begin{cases} 1-t, & t \in [0,1) \\ 0, & t \geq 1 \end{cases}, \quad g(t) = \begin{cases} 1, & t \in [0,1) \\ 0, & t \geq 1 \end{cases}, \tag{4.29}$$

and two, the Gaussian kernel,

$$k(t) = e^{-\frac{t}{2}}, \quad g(t) = e^{-\frac{t}{2}}, \tag{4.30}$$

where $t = \langle \boldsymbol{x}, \boldsymbol{x} \rangle$.

It is a common simplification to see the mean shift operation as shifting a kernel in feature space. However, the kernels are defined around the sample points and evaluated from a shifting coordinate, best illustrated in Eq. 4.23. A bandwidth can be chosen per-sample. The bandwidth controls the resolution of the estimate. It is a crucial parameter of the algorithm that has been investigated in several studies [Coma 03, Wu 07, Huan 08].

Mean shift has key advantages over most other methods. First, other than graph-based segmentation algorithms, clustering can happen in a feature space that is agnostic to pixel coordinates. No topological clues are needed and a segment can consist of several disconnected areas within the image that share a high pixel-wise consistency. For RGB images, pixel coordinates are typically incorporated into the feature space as the color vectors alone do not provide enough information for a global segmentation, but this is not the case for multispectral or hyperspectral data. Second, it has also the important property that no prior information is needed, e.g. a desired number of clusters. Third, mean shift is not bound to a data model, e.g. the data stemming from a specific distributions, or clusters of specific shape or size.

In principle, these properties make mean shift a good choice for clustering of an unseen multispectral or hyperspectral image to aid the user in further exploration without prior knowledge. The major drawback of mean shift however is its computational complexity of $O\left({n_X}^2\right)$, where n_X is the number of data points, or pixels. A quadratic growth in complexity is in fact problematic when analyzing image data of moderately high spatial resolution. In the case of mean shift, it leads to computation times of several hours for a high-resolution image, let alone a multispectral one. In practice, mean shift computation on a whole image of typical spatial and spectral resolution is only feasible through approximations, which we will detail below.

Fast-adaptive Mean Shift

In 2003, Georgescu et al. [Geor 03] introduced the fast adaptive mean shift (FAMS) algorithm which significantly improves computational efficiency by employing locality-sensitive hashing (LSH). In FAMS, the mean shift vector at location $\boldsymbol{y}$ in feature space is defined as

$$\boldsymbol{m}_K(\boldsymbol{y}) = \frac{\sum_{i=1}^{n_X} \frac{1}{h_i^{(n_D+2)}} \boldsymbol{x}_i \, g\left(\left\| \frac{\boldsymbol{y}-\boldsymbol{x}_i}{h_i} \right\|_2\right)}{\sum_{i=1}^{n_X} \frac{1}{h_i^{(n_D+2)}} \, g\left(\left\| \frac{\boldsymbol{y}-\boldsymbol{x}_i}{h_i} \right\|_2\right)} - \boldsymbol{y} \ , \tag{4.31}$$

where h_i is the bandwidth and g is the profile of the Epanechnikov kernel [Geor 03]. The algorithm consists of three steps:

Bandwidth Selection. Unlike the original mean shift, FAMS selects an h_i for each data point determining its radius of influence. Adaptive bandwidths are a crucial step in making the method generalizable, as a common rule for bandwidth selection is not trivial to find. Denoting the n_K-nearest neighbor of $\boldsymbol{x}_i$ as $\boldsymbol{x}_{i,n_K}$, h_i is defined as

$$h_i = \|\boldsymbol{x}_i - \boldsymbol{x}_{i,n_K}\|_1 \ , \tag{4.32}$$

Using the L_1 norm is considered suitable with regard to the characteristics of the LSH data structure explained below. The value n_K needs to be large enough to include a detectable change in density. According to [Geor 03], the selection of parameter n_K is not critical to FAMS, however we need to consider it in Sec. 4.2.2. In general, n_K may be used as a control for the coarseness of the clustering.

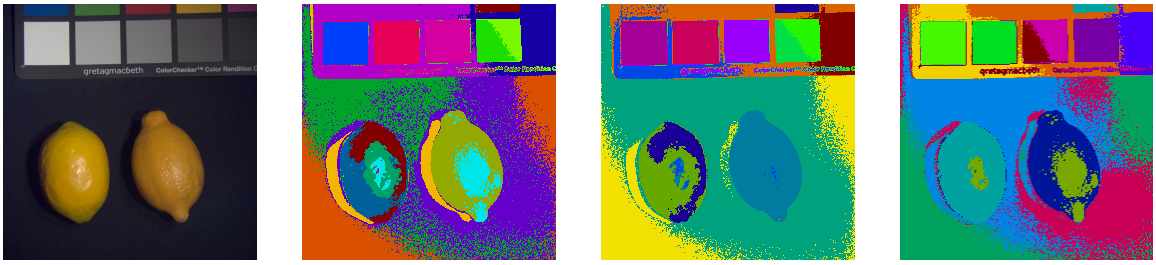
Mean shift. $\boldsymbol{m}_G(\boldsymbol{y})$ is started at each data point and iteratively progresses until it converges, as explained above.

Mode Pruning. In the previous step we obtained a corresponding mode for each data point $\boldsymbol{x}_i$ originating from scene point $\boldsymbol{p}_i$. In the pruning step, common modes of several pixels are identified and merged [Coma 02]. The segmentation consists of a cluster association l_i for each pixel, such that all pixels in a cluster share a common mode $\boldsymbol{\mu}$.

The speedup in FAMS through locality sensitive hashing lies in a reduced complexity by only considering a small subset of the n_X sample points in Eq. 4.31. LSH provides a hash that approximately finds the set of nearest neighbors for a data point. The LSH structure itself is tuned according to the image data. The operations for building and querying the LSH are computationally efficient, especially in relation to the distance calculations performed after each LSH query. The downsides of LSH are tunable parameters K, L that need to be selected, that nearest-neighbor searches can only be performed in L_1, and that the query results are only approximate. Using LSH can therefore degrade segmentation quality. Next to returning a subset of points in each step, the LSH structure also caches the final mode obtained in previous shifts for each query hash. This further reduces execution time by early trajectory termination.

Other Shift-based Algorithms

The introduction of LSH to the algorithm by Georgescu et al. leads to a significant speedup of the mean shift computation. However, on-demand calculation within an interactive usage scenario is still not viable. Recent methods aim at further reducing execution times. The quick shift algorithm by Vedaldi and Soatto can not overcome the $O(N^2)$ complexity [Veda 08]. While variants of the so-called hierarchical mean shift do not tackle the theoretical computation time boundary [Surk 11, Surk 12],



(a) true color **(b)** $K = 30$, $L = 19$ **(c)** $K = 29$, $L = 20$ **(d)** $K = 29$, $L = 19$

Figure 4.17: Median shift results on *Lemons* image. Median shift was run with slight variation in parameters K and L for the locality-sensitive hashing.

they start with smaller search kernels, which in practice reduces the number of data points to be considered in each shift step. Then, they iteratively restart the method with the obtained modes from the previous step as data points. A global segmentation is reached after two to four steps. However, reducing the kernel size becomes increasingly impractical with the high dimensionality of multispectral or hyperspectral vectors.

Freedman and Kisilev propose a fast mean shift variant that employs a sampling step before running mean shift [Free 10]. Therefore, the algorithm only has to be performed on a considerably smaller, random subset of $\mathcal{X}$. Their method however depends on a specific selection of a common bandwidth h. It does not apply to the use of adaptive bandwidths, which in our experience is a crucial ingredient of FAMS for hyperspectral image segmentation.

Median shift by Shapira et al. exploits the LSH aspect of FAMS by taking statistical properties of the LSH as a cue for finding modes in the feature-space density [Shap 09]. While it achieves a significant speed-up, it tends to under-segment the data: modes of less well-represented parts of the scene are not identified and these regions may be falsely attributed to poorly related larger segments. In our experimentation, it was also found that median shift is unstable in regard to the LSH parametrization. Figure 4.17 depicts segmentations obtained with median shift. Small variations in the parameters K and L to the hash generation lead to different partitioning of both lemons and the background.

Shetty and Ahuja proposed the probabilistic shift method, which is based on the detection of isotropic density neighborhoods [Shet 10]. The method addresses the problem of bandwidth selection. Instead of following a local increase in density, the size of an isotropic neighborhood around each point is detected and subsequently used to shift points towards the most influential regions of high isotropy. However we found the method fails to detect larger isotropic neighborhoods in a high-dimensional feature space. Figure 4.18 shows a plot of average influence neighborhood size, corresponding to the size of an isotropic neighborhood, on several images from the *CAVE* dataset. To show the effect of dimensionality, spectra were reduced in dimensionality with a linear filter.

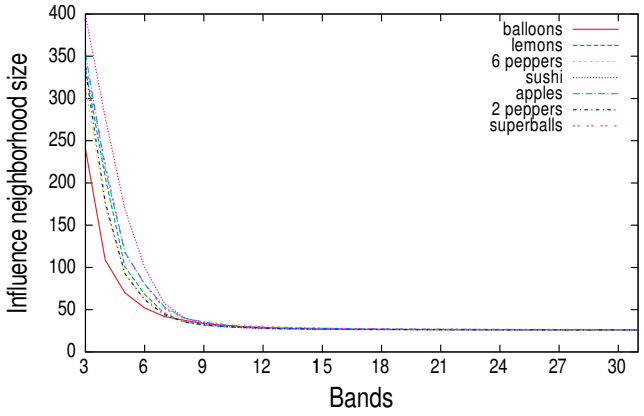


Figure 4.18: Influence neighborhoods of probabilistic shift on a series of images, where spectra were reduced in dimensionality to obtain lesser bands.

SOM-based Clustering

As we learned, the SOM may be used as a versatile tool for various image processing applications. Most often however, it is used as a tool to visualize the data distribution by displaying the map as a so-called U-matrix [Vesa 00b, Ults 03]. In this representation, one can expect clusters in the data to be visually recognizable. Thus, a SOM may also be an interesting tool for unsupervised detection of clusters.

In a single stage approach, the SOM's vector quantization may be seen as a clustering and for this purpose, a SOM is trained with n_M nodes, whereas n_M is the number of desired clusters [Mao 96, Wu 04]. This disregards the topological relationship learned by the SOM. As the vector quantization results of a SOM and k-means are often seen as equivalent in literature when $k = n_M$ [Bac 05], similar results may be obtained by the k-means algorithm. Two-stage approaches train a SOM first, then use it for input to a clustering algorithm. The method of choice is often k-means and typically does not take topology into account. Many algorithms put assumptions on the shape of clusters, which are required to be of either hyperspherical or hyper-ellipsoidal shape, and require the number of clusters to be known a priori [Wu 04, Jian 04, Dong 05, Orti 13]. Next to k-means, Vesanto et al. propose a hierarchical clustering on the model vectors, which is based on a criterion for inter-cluster distance, also ignoring map topology [Vesa 00a]. The benefit of the SOM stage in this scenario, when discussed from the perspective of hyperspectral image analysis, is questionable as k-means clustering without such a preprocessing step is computationally efficient enough to be run directly on the input data. Note that the vector quantization alone does not provide a means for clustering methods to operate in a lower dimensional space, i. e. on the data manifold. Instead they only work on a sparsified representation of the data distribution.

Taşdemir et al. apply agglomerative hierarchical clustering on the SOM based on the concept of the Topographic Error (see Eq. 3.37 on page 51) [Tasd 11]. The connectivity of a pair of neurons $\boldsymbol{m}_k$ and $\boldsymbol{m}_l$ is defined by the sum of pixels $\boldsymbol{x}$ for which $\boldsymbol{m}_k$ is the best matching unit and $\boldsymbol{m}_l$ is the second-best matching unit, or vice-versa. In each iteration, the pair of most similar clusters according to the pairwise connectivity of their respective members is then merged. Merging commences until

only one cluster is left; the level in the hierarchy used for the final clustering (i. e. the number of clusters) needs then to be known a priori or determined based on general statistics [Tasd 11]. Note that this clustering also ignores the topology of the SOM, although it could be incorporated by restricting the set of cluster pairs allowed to merge.

A clustering algorithm that takes map topology into account was proposed by Wu and Chow [Wu 04]. It also performs agglomerative hierarchical clustering of the model vectors. Topology plays a role as only clusters that contain neighboring neurons can be merged. Merge criterions are based on a cluster validity index which takes both a cluster's compactness (similar to Eq. 4.22) and separation from other clusters into account. The index is computed based on all input samples that, through their corresponding BMU, belong to each cluster. Before clustering is performed, the method needs an extensive pre-processing routine that prunes feature vectors and neurons according to heuristic criteria to be less affected by noise and outliers in the data [Wu 04].

4.2.2 Mean Shift on Superpixels

As discussed, we consider the mean shift clustering approach to be a good fit for the challenge of finding clusters in hyperspectral data. It is effective in high dimensions [Geor 03] and does not put constraints on the shape or size of clusters to be found. However, the high spectral resolution and increasingly high spatial resolution of hyperspectral images renders direct calculation of mean shift, or the more efficient FAMS, unsuitable for interactive analysis. One solution to the problem is a significant reduction in the input data. We expect all pixels from a small, homogeneous image region to fall into the same FAMS cluster. Instead of finding a cluster for each pixel, we could bundle the effort for all pixels in such a small homogeneous region. Superpixel segmentation helps us to identify such regions. This work was first published in [Jord 13b].

While our method is agnostic to the superpixel segmentation method used, we chose the algorithm by Felzenszwalb and Huttenlocher (FH04) as detailed in Section 4.2.1. FH04 was shown to be effective in the domain of hyperspectral endmember detection [Thom 10]. It has several advantageous properties that are particularly well-suited for our application. It is crucial for us that boundaries in the image are not missed and superpixels stay confined within a homogeneous region with high intra-similarity. FH04 achieves the highest boundary recall in a benchmark of several superpixel methods on the Berkeley Dataset [Acha 12, Arbe 11]. Other superpixel methods often fulfill different properties, e.g. higher regularity in shape or size, that are of no concern to us. Also, FH04 has a time complexity of $O(n_X \log n_X)$. In the aforementioned benchmark, it ranks second in computation speed. This makes it an ideal choice for our interactive setting. The parameter c manipulates the degree to which the difference between two superpixels must be greater than their internal differences to favor a split. We configure FH04 to serve as a pre-processing step in clustering by setting a low c value, which decreases the average superpixel size.

Hyperspectral Superpixels

A key component of graph-based algorithms like FH04 is the setting of $w_{i,j}$, the weight of each edge in the graph. In the original implementation, $w_{i,j} = |x_i - x_j|$, where x_i is the intensity value of a grayscale pixel. For RGB images, the authors run the algorithm on each band separately [Felz 04]. A weighted Euclidean distance was proposed for four band RGB+NIR images [Cui 10]. Both adaptations are not applicable to high-dimensional multispectral images.

A reasonable solution is to employ similarity measures specifically designed for spectral data. Recall the discussion of established spectral matching measures in Section 2.4.2 (page 21). The Spectral Angle (SA) and the Spectral Information Divergence (SID) are believed to best discriminate different materials based on their characteristic spectra. SA has already been used in conjunction with FH04 on remote sensing data [Thom 10]. We will evaluate several similarity measures including SA and SID for another graph-based segmentation algorithm in Section 4.3.3. For this algorithm, we evaluate SA (Eq. 2.11) and SID (Eq. 2.12).

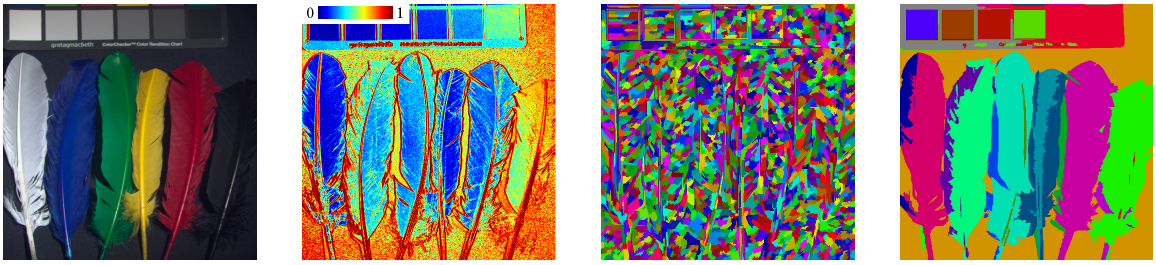
The ratio between weights is used when deciding whether to merge superpixels. We empirically found that the highly non-uniform distribution of SID on our test images poses a problem. Therefore, we apply histogram equalization on the edge weights as explained in Section 2.4.2 (page 23). As a side effect, this procedure produces integer values, effectively reducing FH04 complexity [Felz 04]. Since histogram equalization improves the results for SA as well, we make it a fixed part of the algorithm.

Superpixel Mean Shift

We propose two new algorithms which combine multispectral superpixels with FAMS. The idea behind both methods is to significantly reduce the amount of input data, yet to maintain sufficient detail for obtaining a good segmentation quality. Depending on how this is done, remarkable speed-ups can be achieved.

Per-superpixel Mean Shift (PSPMS) In this variant, we sacrifice spatial detail for execution speed. Superpixels $\mathcal{S}_j, j \leq n_S$ are computed on the original image. The feature space however is unchanged, i.e. it may consist of the n_X spectral vectors $\boldsymbol{x}$ or of their representation in another feature space, e.g. the spectral gradient. For each $\mathcal{S}_j$ we compute the centroid $\boldsymbol{s}_j$ of all data points $\boldsymbol{x}_k \in \mathcal{S}_j$. An adaptive bandwidth is selected for each pixel $\boldsymbol{x}$. However, instead of starting the mean shift procedure at each $\boldsymbol{x}_i$, we start it at each $\boldsymbol{s}_j$. In all other aspects, PSPMS runs like FAMS. As a last step, the cluster assignment l_j of $\mathcal{S}_j$ is back-projected to all $\boldsymbol{x}_i \in \mathcal{S}_j$. We obtain a full segmentation.

In our experiments we observe a fivefold to tenfold speed increase as compared to FAMS. Algorithmic complexity is only reduced by a constant factor, as the adaptive bandwidth selection still has a complexity of $O({n_X}^2)$. The mean shift procedure however is considerably more time consuming than bandwidth computation due to its iterative nature, and that is reduced to $O({n_S}^2)$. Segmentation results of this variant best mimic FAMS behavior, as the superpixels only affect the spatial resolution, but not the feature space.



(a) image in true color **(b)** edge weights in x-direction **(c)** superpixel segmentation **(d)** PSPMS segmentation

Figure 4.19: Illustration of PSPMS on *Feathers* image. Superpixels in (c) are computed from edge weights exemplified in (b) (weights in y-direction not shown). Using (c) for starting points results in the global clustering seen in (d).

Full Superpixel Mean Shift (FSPMS) In this variant, we fully leverage the superpixel data representation. The centroids $\boldsymbol{s}_j$, $j \leq n_S$ form the feature space. Adaptive bandwidths are computed per superpixel. Then, from each $\boldsymbol{s}_j$ the mean shift is performed. Cluster assignments are back-projected to the pixels after mode pruning. To achieve good results, we need to alter the algorithm in bandwidth selection and mean shift steps.

Bandwidth Selection. The bandwidth for each data point is chosen so that n_K nearest neighbors lie within its bounds. Georgescu et al. reason that the choice of n_K is not critical for the performance of the algorithm [Geor 03]. However, the adaptive bandwidths are directly related to n_K. If the feature space is sparsely populated, n_K has a strong effect on bandwidth size. As a result, data points influence a higher number of shift trajectories, leading to fewer distinct modes. While typical input images have 2^{18} pixels (in our test images) or more, a superpixel segmentation on these images produces between 2^{11} and 2^{14} superpixels. Therefore a n_K suitable for FAMS or PSPMS is not suitable anymore for FSPMS and would yield a broad under-segmentation. From our experiments we derive an effective rule for the choice of n_K: $n_K = p \cdot \sqrt{n_S}$, whereas n_S is the number of input samples, in our case, superpixels. The linear factor p is a tunable parameter that influences the coarseness of the segmentation. Georgescu et al. also suggest to take the feature space dimensionality into account when choosing n_K.

Mean shift. The key idea behind the mean shift algorithm is the estimation of the density gradient. Our new feature space is based on the rationale that the superpixels provide a good sparse representation of the image's distribution. This is only the case when each superpixel both represents a homogeneous region in the image, and has the same weight as the points that are represented by it. Therefore, we weight the bandwidth of each superpixel centroid by the superpixel size.

The main advantage of this variant is that complexity is reduced to $O(n_X \log n_X + {n_S}^2)$. This makes the method viable for interactive applications, where a result should be obtained within a few seconds.

Figure 4.19 illustrates the operation and results of PSPMS on an example image. From the input image, SID edge weights are calculated including histogram

equalization. FH04 is run with parameter $c = 0.25$ to obtain a rather coarse early segmentation. PSPMS is then run with superpixels as input.

In the superpixel-based acceleration of FAMS, a viable compromise is found between computational complexity and spatial resolution of the resulting segmentation. However, this works best for images with a considerably high spatial resolution in relation to the depicted objects. While spatial features are steadily gaining more traction in the field of remote sensing, e.g. for improving classification performance [Fauv 13], or for reducing complexity in spectral unmixing [Shi 14], the effective use of superpixels is still limited in many exemplary images from this domain. The same limitation can be found in images depicting natural scenes, e.g. from the *Foster* dataset.

4.2.3 SOM-based Clustering

Due to the aforementioned cutback of the sparsification in the spatial domain, we now propose two methods that use a Self-organizing Map for efficient global clustering. In the first approach, we incorporate the SOM into the FAMS algorithm. Our goal is to match or approximate the solutions given by FAMS when applied on the same input image. In the second approach, we establish a clustering method based on the topological relationship of neurons in the map.

SOM-aided Mean Shift

The FAMS algorithm reduces the complexity of the shift operation by approximating the nearest-neighbor search. Effectively, the hashing of each data point and subsequent early trajectory termination based on the hash leads to a quantization of the feature space. This shows that it is viable to reduce starting points of the algorithms by exploiting knowledge about the mode associated with another data point in close proximity. In a similar spirit, Freedman and Kisilev reduce complexity by reducing the sample set altogether [Free 10]. While their random selection of samples is tied to a fixed bandwidth choice, a more representative smaller set of samples might be capable of representing the data distribution well-enough without such constraints. As was seen by example of the median shift, the LSH structure itself likely is not fit to robustly generate such a representation.

We investigate a different take at reducing the input to mean shift in the feature space. As explained in Section 4.2.1, mean shift is a density gradient estimator. In Chapter 3 we discussed manifold learning methods to find a sparse representation of the sample distribution. The self-organizing map in particular was shown to well-capture the original data distribution in a quantized way. We suggest to apply mean shift on said sparse representation to reduce complexity. Recall the definition of the mean shift vector from Eq. 4.31,

$$\boldsymbol{m}_{\kappa}(\boldsymbol{y}) = \frac{\sum_{i=1}^{n_X} \frac{1}{h_i^{(n_D+2)}} \boldsymbol{x}_i \, g\left(\left\| \frac{\boldsymbol{y}-\boldsymbol{x}_i}{h_i} \right\|_2\right)}{\sum_{i=1}^{n_X} \frac{1}{h_i^{(n_D+2)}} \, g\left(\left\| \frac{\boldsymbol{y}-\boldsymbol{x}_i}{h_i} \right\|_2\right)} - \boldsymbol{y} \,, \tag{4.33}$$

where $\mathcal{X}$ is the set of all available samples, in our case all pixels. We can represent each pixel $\boldsymbol{x}$ by its best-matching unit, $\boldsymbol{m}_c$. The mean shift vector is then given as

$$\boldsymbol{m}_\kappa(\boldsymbol{y}) = \frac{\sum_{k=1}^{n_M} \frac{n_{Y_k}}{h_k^{(n_D+2)}} \boldsymbol{m}_k \, g\left(\left\|\frac{\boldsymbol{y}-\boldsymbol{m}_k}{h_k}\right\|_2\right)}{\sum_{k=1}^{n_M} \frac{n_{Y_k}}{h_k^{(n_D+2)}} \, g\left(\left\|\frac{\boldsymbol{y}-\boldsymbol{m}_k}{h_k}\right\|_2\right)} - \boldsymbol{y} \,, \tag{4.34}$$

where n_{Y_k} is the amount of pixels in $\mathcal{X}$ represented by $\boldsymbol{m}_k$, that is, $\boldsymbol{m}_k$ is the BMU of said pixels. The bandwidth h_k is regularly computed in the bandwidth selection step of FAMS, using $\mathcal{M}$ as data points.

Effectively, each pixel is approximated by its BMU. To find the mode for $\boldsymbol{x}$, we start the mean shift operation at $\boldsymbol{m}_c$. The result is then shared between all pixels represented by $\boldsymbol{m}_c$. This reduces the complexity of the algorithm, including SOM training and BMU lookup, to $O\left(n_X n_M + {n_M}^2\right)$. Locality-sensitive hashing which is typically a part of FAMS becomes unnecessary for efficient computation. It is not employed to avoid the additional inaccuracy it would impose. This method is denoted as `mssom` in our experiments.

SOM Topology Segmentation

We note that the previous approach exploits the capability of the SOM to find a representative set of model vectors to describe the original data distribution. However, it does not exploit the topological information the SOM provides on the distribution. We can formulate two assumptions on the topology preservation in the SOM. One, we can assume that neurons $\boldsymbol{m}_k$ and $\boldsymbol{m}_l$ are close in the feature space if their locations $\boldsymbol{r}^{(k)}$ and $\boldsymbol{r}^{(l)}$ are in close proximity with each other in the map. This means that the geodesic distance between $\boldsymbol{r}^{(k)}$ and $\boldsymbol{r}^{(l)}$ is well-approximated by the Euclidean distance. Two, a cluster in the high-dimensional distribution is represented by a cluster in the SOM, i. e. a connected component in the map.

Based on these properties, we apply the method by Felzenszwalb and Huttenlocher on the SOM. The graph $(\mathcal{V}, \mathcal{E})$ is defined by the SOM lattice. The edge weight between neurons $\boldsymbol{m}_k$ and $\boldsymbol{m}_l$ is given by $\|\boldsymbol{m}_k - \boldsymbol{m}_l\|_2$. Initially, each neuron $\boldsymbol{m}_k$ represents a component, corresponding to a single stage SOM clustering. Then, the second stage consists of iteratively merging components based on the pairwise comparison predicate given by Eq. 4.17 (page 86). To account for neurons that represent the transition between two clusters in the SOM, we enforce a minimum component size of $0.02 n_M$. Otherwise, transitional model vectors would often form distinct clusters that do not represent a specific material or object in the scene.

Then, the cluster label l of each $\boldsymbol{x}$ is set to the label of $\boldsymbol{m}_c$, whereas $\boldsymbol{m}_c$ is the BMU for $\boldsymbol{x}$. We obtain a segmentation that has connected components in the SOM, but is global in the image domain. As the clustering is performed on model vectors, its time complexity is $O\left(n_M \log n_M\right)$, i. e. constant in n_X. The upper bound in overall time complexity of $O\left(n_X n_M\right)$ is caused by the final pixel labeling (which can be highly parallelized).

As was noted in the related work, Wu and Chow incorporate map topology in a hierarchical clustering scheme that is based on statistics for inter-cluster and intra-cluster density [Wu 04]. Our method has several advantages over Wu and Chow.

First, our clustering is performed independently of the image data. Measures for merging clusters are derived solely on the SOM model vectors. This is advantageous in computational complexity as well as algorithmic complexity. Second, Wu and Chow only use the topology for a general selection of which components can be merged. In our method, the direct relationship of neurons at the border of two clusters is used to measure the difference between components. Third, our method does not rely on any preprocessing or heuristics for removing neurons from consideration. Instead, it is by design robust towards outliers and noise. This method is denoted as `somtopo` in our experiments.

Fuzzy Clustering

Both our methods calculate a SOM in a two-stage approach to clustering, where the image segmentation is achieved by mapping cluster assignments l_k back to pixels associated with $\boldsymbol{m}_k$. This association is defined by Eq. 3.9. However, as discussed in Section 3.2.2, we can gather more information about a pixel with a multi-BMU lookup. This also holds in the case of clustering.

Consider a pixel $\boldsymbol{x}$ that contains a mixture of several prototype spectra contained in the scene. Such pixels are likely to occur in remote sensing images with a low spatial resolution. They are also found in the case of inter-reflectance, i. e. irradiance on an object caused by the reflection of another object, or at the border between two objects. The BMU array of these pixels typically contains BMUs from several clusters. We can find these clusters and approximate their contribution to the mixture through rank weighting.

The contribution of the cluster with label l to $\boldsymbol{x}$ is then seen as

$$\sum_{k=1}^{n_C} [l_{\boldsymbol{c}_k} = l] \cdot w_k \,, \tag{4.35}$$

where w_k is obtained via Eq. 3.18 or Eq. 3.19 (see page 38) and $\sum_{k=1}^{n_C} w_k = 1$.

4.2.4 Experimental Results

When evaluating segmentation algorithms on multispectral and hyperspectral images, we face the same challenges as in the case of edge detection. Ground-truth data is not publicly available for any dataset of our domain to date. Creating such data is hard: The mere definition of which boundaries in an image are relevant or not, or likewise, which clusters should be split into two, is often subjective. The Berkeley Segmentation Data Set is a great example for such a benchmark for color images, which also shows the necessary amount of work [Arbe 11]. Each image is annotated by several subjects, resulting in different levels of detail for the marked contours.

Unsupervised quantitative measures were proposed for clustering evaluation when ground-truth segmentations are lacking. We mentioned earlier the compactness measure C computed by Eq. 4.22 on page 87. It has the shortcoming of assuming hyperspherical clusters that share a diameter. Several other proposed measures work only for monochromatic images [Chab 06]. Zhang et al. propose an

entropy-based measure [Zhan 04]. Computation of entropy needs to be done band-wise, is generally problematic for a higher dynamic range and even less applicable to the continuous normalized or spectral gradient feature spaces. Corcoran et al. introduce a measure for object-based analysis of remote sensing images [Corc 10]. It is inspired by the human vision system, designed for segmentation in the spatial domain, and not applicable to a global clustering. Other measures that were applied in the hyperspectral domain rely on the compactness, with an additional penalty on the number of segments to counter the effect that over-segmentation profits from the compactness measure [Zhan 12]. In short, we are restricted in capabilities of quantitative evaluation and in most cases will rely on qualitative evaluation and general statistics.

In some of the experiments, our focus lies in how the results of tested algorithms differ from FAMS output. We use three statistics in judging the segmentations in this regard. First is the number of obtained segments. In general, the segmentations of FAMS provide enough detail without adverse over-segmentation. We expect to obtain a similar number of segments as with FAMS. Second is the compactness C as mentioned above. While it can be a misleading measure of segmentation quality, it should be comparable across methods in our case, and we report it relative to FAMS. Third is the Rand index [Rand 71]. It measures the similarity between two data clusterings. For the set of pixels $\mathcal{X}$ we have corresponding sets of reference labels $\mathcal{K}$ (computed via FAMS) and obtained labels $\mathcal{L}$, respectively. For each pair of pixels $\boldsymbol{x}_i$, $\boldsymbol{x}_j$, we have corresponding labels l_i, l_j and k_i, k_j. The Rand index is given by

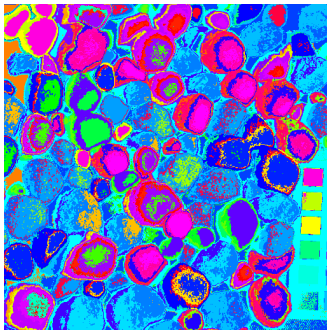
$$R = \frac{\sum_{\boldsymbol{x}_i \in \mathcal{X}} \sum_{\substack{\boldsymbol{x}_j \in \mathcal{X} \\ i \neq j}} [l_i = l_j, k_i = k_j] + [l_i \neq l_j, k_i \neq k_j]}{\binom{n_X}{2}}, \tag{4.36}$$

which is the fraction of disjoint pixel pairs for which $\mathcal{L}$ agrees with $\mathcal{K}$. Perfectly matching clusterings achieve $R = 1$. On images from the *CAVE* dataset, we mask pure, non-shadowed background pixels from the computation of R, as they often constitute a major part of the image, are particularly noisy and could undermine the informative value of the Rand index.

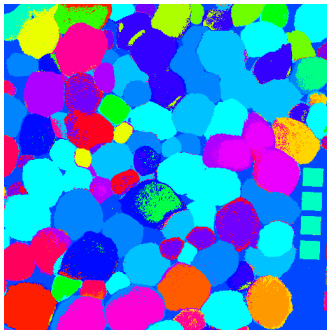
We depict clustering results as color maps, whereas each label l is assigned a distinct color. Colors are randomly selected in the hue-saturation-value (HSV) color space. We divide the hue range equidistantly according to the number of clusters, while saturation and value components are set to 100 %. Black may be used as an additional color. In some cases, for better discrimination, we also add a random variety to the value component. Note that this is not done when spectral plots are shown, as the parallel coordinate plotting relies on a transparency component, which works best with a fixed value component. For more details see Section 5.3.1.

Our implementation of the proposed mean shift-based methods is a heavily modified version of MultiSFAMS [Ange 05]. We added support for higher-resolution images, parallelization of the adaptive bandwidth calculation, and Single Instruction, Multiple Data (SIMD) acceleration for vector distance computations. We avoided

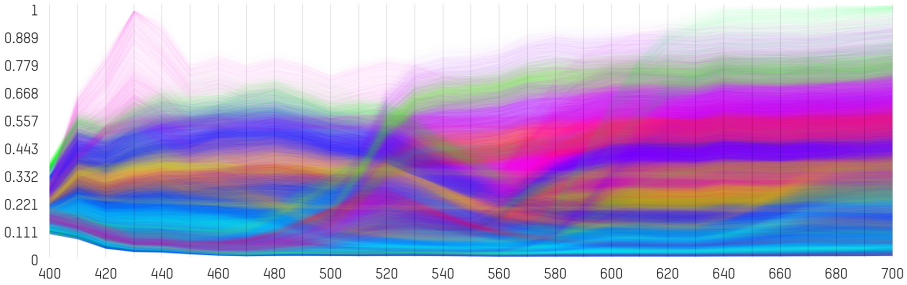
(a) true color



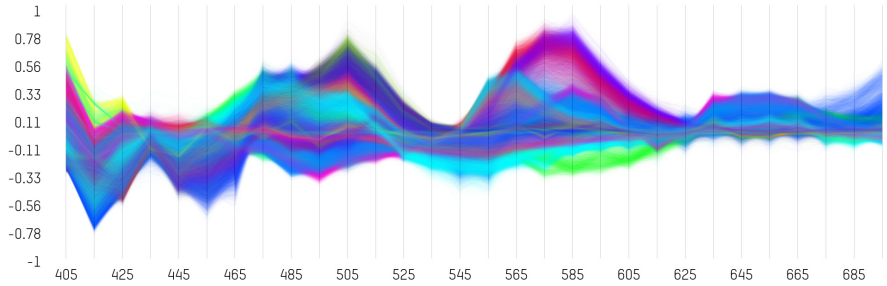
(b) FAMS



(c) SG-FAMS



(d) spectral distribution, colored by FAMS



(e) spectral gradient distribution, colored by SG-FAMS

Figure 4.20: Feature space comparison for clustering on *Pompoms* image. Please refer to page 52 for an explanation of spectral distribution plots.

parallelizing the shift step as it interferes with early trajectory termination. Likewise, our implementation of FH04 is based on the source code released by the authors [Felz 04].

Feature Space Selection

Figure 4.20b shows a segmentation obtained by applying our FAMS version on the *Pompoms* image from the *CAVE* dataset depicted in Figure 4.20a. At first glance, the result does not look satisfactory. However, further investigation reveals that the algorithm does in fact find distinguished clusters in the underlying distribution, as can be seen in Figure 4.20d. Clusters are mostly driven by the overall magnitude of spectral vectors. Geometric effects are very dominant in the original image feature space, as discussed in Section 2.3. The segmentation distinguishes brightness

Algorithm	c	Segments	ΔC (1%)	R	Time (1s)
SG-FAMS		22.7 ± 7.1			629.3 ± 262.0
SG-FAMS*		−1.0 ± 2.7	−0.5 ± 7.3	0.96 ± 0.04	
PSPMS	0.05	+16.3 ± 27.1	−1.1 ± 3.8	0.96 ± 0.03	113.2 ± 39.9
	0.25	+14.8 ± 20.2	+1.7 ± 4.9	0.95 ± 0.04	79.6 ± 32.8
FSPMS	0.05	+2.4 ± 8.5	−2.5 ± 5.7	0.94 ± 0.03	9.0 ± 1.5
	0.25	−3.3 ± 8.7	+3.8 ± 11.0	0.88 ± 0.11	5.9 ± 1.1

Table 4.1: Statistical results averaged over nine test images.

variations on objects of uniform material. Different materials however are often missed. In this example, pompoms of different colors fall in the same clusters.

To overcome this issue, we employ a different feature space, in similar fashion to the use of the $L^*u^*v^*$ color space as opposed to RGB [Coma 02]. The spectral gradient descriptor by design separates material information from reflectance content. Hence, in order to get a mean shift segmentation that focuses on material properties we use SG-FAMS, a spectral gradient variant of FAMS. In SG-FAMS we first compute the spectral gradient from each pixel $\boldsymbol{x}$ (see Eq. 2.3, page 20), then use the gradient image as input to the FAMS algorithm. An alternative for lessening the effect of geometry is the L_2-normalized feature space (see Eq. 2.2, page 19). It is employed by Huynh and Robles-Kelly for material clustering [Huyn 10b].

Figure 4.20c shows the segmentation obtained by SG-FAMS on the same image. We observe a smaller amount of segments that better cover the different pompom colors. Different materials fall into the same segment in rare cases. This is also reflected in the spectral gradient distribution plot in Figure 4.20e.

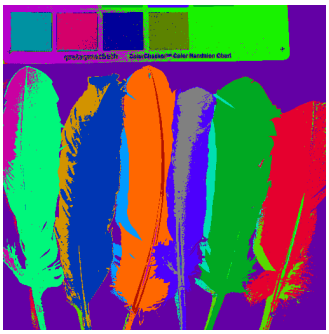
Superpixel Meanshift

We investigate our proposal for a spatial approximation of SG-FAMS using superpixels, namely the per-superpixel mean shift (PSPMS) and full superpixel mean shift (FSPMS) algorithms. Both the SA and the SID similarity measures were initially tested for edge weights w_{ij}. Both work well when histogram equalization is applied. Most often, the segmentations obtained with SID are on-par or of higher quality. We, therefore, only report results computed with SID. We present two different settings of c to judge the influence of superpixel size. We keep the FAMS parameters fixed to $p = 1$ for the adaptive bandwidth selection, $K = 20$, $L = 10$ for LSH [Geor 03].

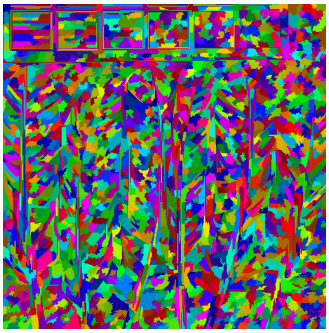
All methods were run five times on an Intel Core i7-2600 CPU with eight threads. For $c = 0.25$, about 2350 superpixels were obtained on each image, with a standard deviation between images of 230. For $c = 0.05$, about 16700 superpixels with a standard deviation of 1180 were obtained. Table 4.1 lists the number of final segments and execution times for each method. The high standard deviation for FAMS execution times is explained by varying opportunities for early trajectory termination in the different images. Furthermore, it lists the difference in compactness, ΔC, when compared to SG-FAMS in percent, and the Rand index with SG-FAMS as a reference. A low value for ΔC, preferably negative, and a high value for R are

(a) image in true color

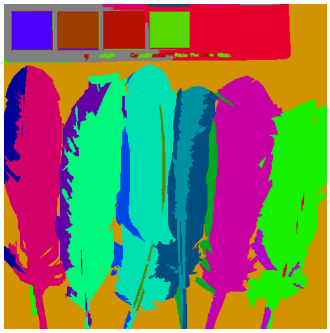
(b) hand-labeled objects



(c) SG-FAMS, 744.9 s

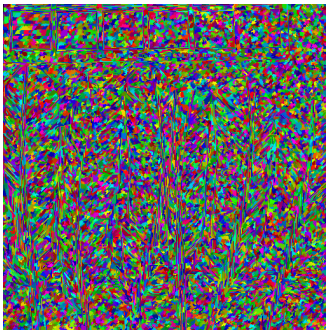


(d) superpixels, $c = 0.25$

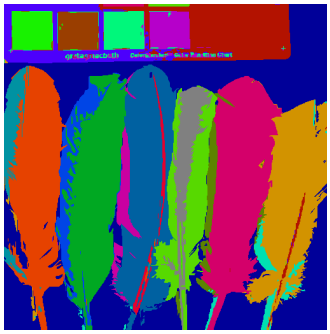


(e) PSPMS on (d), 82.4 s

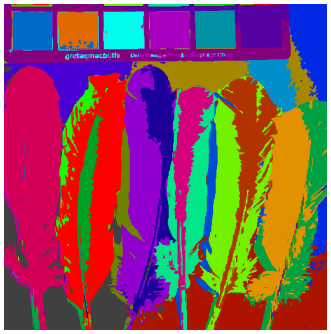
(f) FSPMS on (d), 5.7 s



(g) superpixels, $c = 0.05$

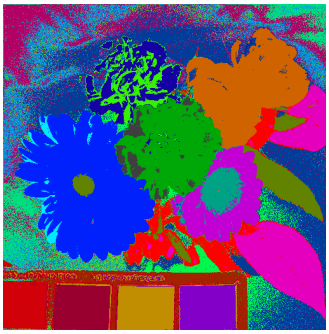


(h) PSPMS on (g), 116.4 s

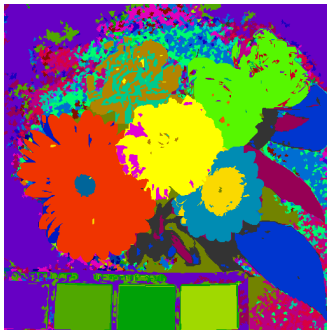


(i) FSPMS on (g), 9.1 s

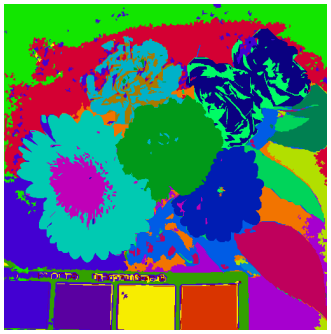
Figure 4.21: Segmentation results on *Feathers* image. Execution times after image loading are denoted next to algorithm names.



(a) SG-FAMS, 407.5 s



(b) PSPMS, $c = 0.05$, 120 s



(c) FSPMS, $c = 0.05$, 8.1 s

Figure 4.22: Segmentation results on *Flowers* image (see Figure 3.2 on page 33).

desired. The listed method SG-FAMS* is a second run of SG-FAMS with a different random initialization for LSH. It shows the variation in these numbers that should be expected. Individual results on the nine images are listed in Appendix D on pages 176–177.

While FSPMS obtains a similar amount of segments as SG-FAMS, PSPMS tends to a slight over-segmentation. The majority of the spike in segment numbers however can be attributed to a failure in appropriately merging superpixels in the noisy background of the *Egyptian Statue* image. We will discuss this image more closely on page 104. Another image that led to a considerable discrepancy in segment numbers is *Flowers*, where these segments also occur in a background region (see Figure 4.22b). From the numbers ΔC and R, we see that a combination of FSPMS with a small amount of superpixels leads to a significant degradation in how well the SG-FAMS result is replicated. Both PSPMS and FSPMS fare well in regards of ΔC and R with the smaller superpixels. The achieved speedup is quite considerable.

In Figure 4.21, the relationship between average superpixel size and PSPMS, FSPMS results is illustrated. Figure 4.21b shows a hand-labeling from [Jord 12b] that helps in judging the segmentation quality. We can see that superpixel size has less effect on the result of PSPMS than FSPMS, whereas in the latter case, it influences the overall characteristic of the segmentation. In this image, FSPMS has the tendency to generate slightly more segments, especially in the background. A challenging example is the *Flowers* image as depicted in Figure 4.22. A good segmentation of all six flowers is hindered by shading effects. PSPMS mostly differs from SG-FAMS in how it handles the noisy background. In general, both SG-FAMS and derived methods have a tendency to over-segment the background. FSPMS puts emphasis on different aspects. All methods successfully capture the flowers as well as the leaves in the background. An example FSPMS segmentation on another dataset can be found in Appendix D on page 178.

We observe that on our data, PSPMS provides a reasonable speed-up without evident loss in segmentation quality. Another advantage of PSPMS over SG-FAMS in many, but not all images is that it is more resistant to noise on the pixel level. FSPMS segmentations are not always on par with the other methods, as some details are missed in comparison. However, they prove functional and can easily be further refined in an interactive setting.

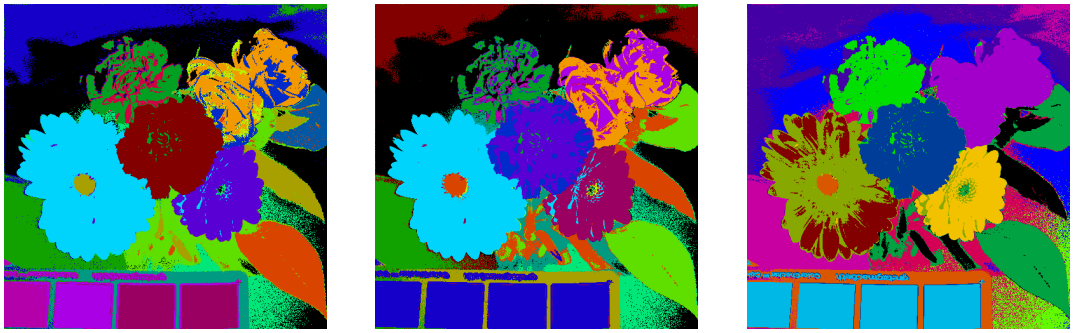
SOM-based Clustering

In the same vein as in the previous test, we now consider our quantization in the feature space, `mssom` as an approximating alternative to SG-FAMS, and our topology-based SOM clustering method `somtopo`. In the mean-shift based methods, we use the same rule for selection of adaptive bandwidths and otherwise parameters as in the case of FSPMS. In the case of `mssom` with $n_M = 1000$, we set parameter $p = 0.65$. For `somtopo`, we train 2-D SOMs and define $c = 0.1$. This test has been carried out on an Intel Core i5-6600 CPU with four threads.

Table 4.2 shows statistical results for these methods on the same dataset as in the previous test. As we found that FAMS works better with a higher number of feature vectors, we trained a larger SOM of size $n_D = 20^3$ as well as a regularly sized map. The result for this variant of `mssom` matches the original result closely. With

Algorithm	n_M	Segments	ΔC (1%)	R	Time (1s)
SG-FAMS		22.7 ± 7.1			553.5 ± 350.5
`mssom`	8000	1.3 ± 5.0	−6.0 ± 11.1	0.93 ± 0.04	129.7 ± 8.5
	1000	−8.2 ± 6.3	−0.1 ± 10.6	0.91 ± 0.05	10.5 ± 0.2
`somtopo`	4096	−6.8 ± 7.4	0.0 ± 13.8	0.90 ± 0.04	13.3 ± 0.6
	1024	−7.8 ± 7.5	−0.5 ± 13.1	0.90 ± 0.05	3.4 ± 0.1

Table 4.2: Statistical results averaged over nine test images.



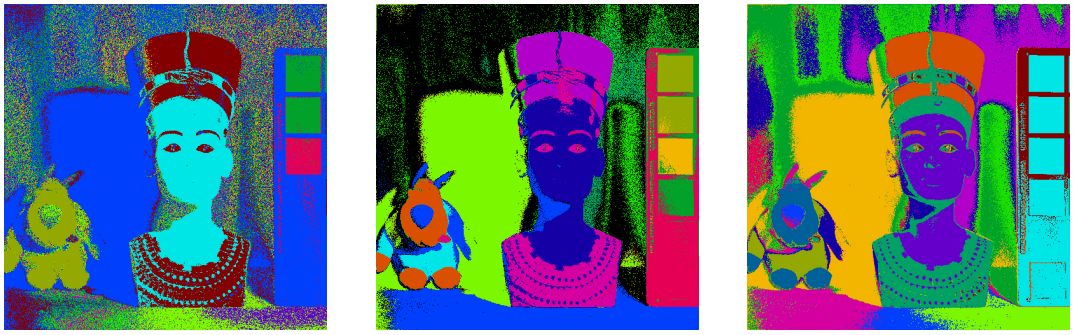
(a) `mssom`, $n_M = 10648$ **(b)** `mssom`, $n_M = 1000$ **(c)** `somtopo`, $n_M = 1024$

Figure 4.23: Segmentation results on *Flowers* image. For SG-FAMS see Figure 4.22a.

$n_D = 1000$, less clusters are formed. As `somtopo` is not a mode-seeking algorithm, we can expect less agreement with the SG-FAMS result in general. The obtained numbers for C and R hint at a similar clustering quality. We also see that for this method, a larger map comes with no general benefit. When also taking execution time into account, `somtopo` appears quite strong in this comparison. Individual results on the nine images are listed in Appendix D on pages 176–177.

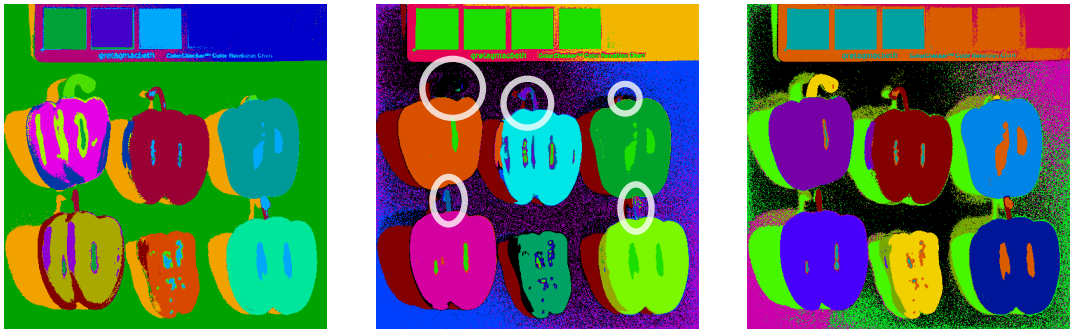
Figure 4.23 shows results on the *Flowers* image, complementing Figure 4.22. We can see that the `mssom` methods differentiate the flower on the top right, similar to FSPMS, where SG-FAMS and `somtopo` only produce one segment. `somtopo` shows more details on the bottom-left flower. On this image, the methods presented in Figure 4.23 show less noisy segments in the background than SG-FAMS and also less background segments than the superpixel methods. This is not a general rule, in some cases we see the opposite.

In Figure 4.24, we see a failure case of SG-FAMS. See Figure 4.9a on page 74 for a true-color display of the image, and Figure D.9 on page 178 for a more revealing false-color visualization. The Nofretete statue can be considered under-segmented, some areas of the image, including on objects, lead to a noisy over-segmentation and some object parts share a cluster with background regions. The SOM-based methods fare better here, omitting a noisy over-segmentation and preserving more details on the stuffed toy, which also results in a significantly improved compactness. While `somtopo` is not a shift-based algorithm, it has similar problems as the shift-based methods. It performs slightly worse than `mssom` with a large map size, and on-par with `mssom` on a similar map size (not shown here), with the benefit of a very fast computation.



(a) SG-FAMS **(b)** `mssom`, $n_M = 10648$ **(c)** `somtopo`, $n_M = 1024$

Figure 4.24: Segmentation results on *Egyptian Statue* image. True color on page 74.

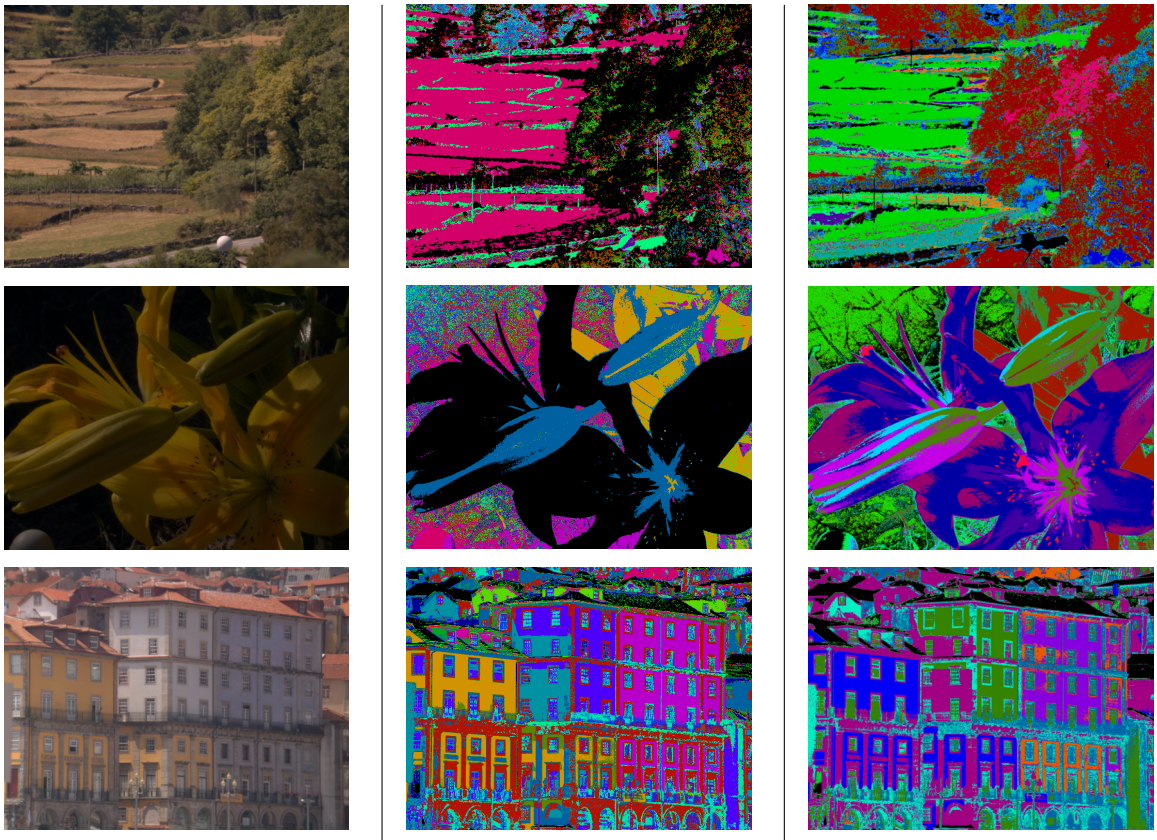


(a) SG-FAMS **(b)** `mssom`, $n_M = 1000$ **(c)** `somtopo`, $n_M = 1024$

Figure 4.25: Segmentation results on *Fake and Real Peppers* image. Marked regions in (b) show pepper stems, as explained in the text. True color on page 15.

Figure 4.25 shows results on the *Fake and Real Peppers* image depicted in Figure 2.6a on page 15. The SG-FAMS segmentation shows a clean result that captures the six different pepper materials, shadows and specular highlights in different segments. We also observe strong inter-reflections from the peppers to the left towards the peppers in the middle, which fall into the segments of the originating materials. Here, we display the result of the smaller SOM in Figure 4.25b to show a deficit in its segmentation. While all peppers, fake and real, fall into their respective segments, the stems of the real peppers share a segment with the background. Furthermore, the green stems of the fake peppers fall in different segments than the body of the green plastic pepper. Both effects are marked in white in Figure 4.25b; a more detailed explanation of the image can be found in Figure 5.18 on page 145. On this image, the behavior of `somtopo` is more similar to SG-FAMS. A distinction is how the background is handled. `somtopo` finds more segments, which are in general not expected, however they also reveal secondary shadows (lime green) next to the primary shadows (strong green), disclosing that a second light source was present when the scene was captured.

We have seen that most segmentations presented in this experiment are of similar quality according to statistics. However, to assess clustering quality in a stringent way, the individual segmentation results need to be studied in detail. Our observation on this dataset is that, while not perfect, the SG-FAMS method in general



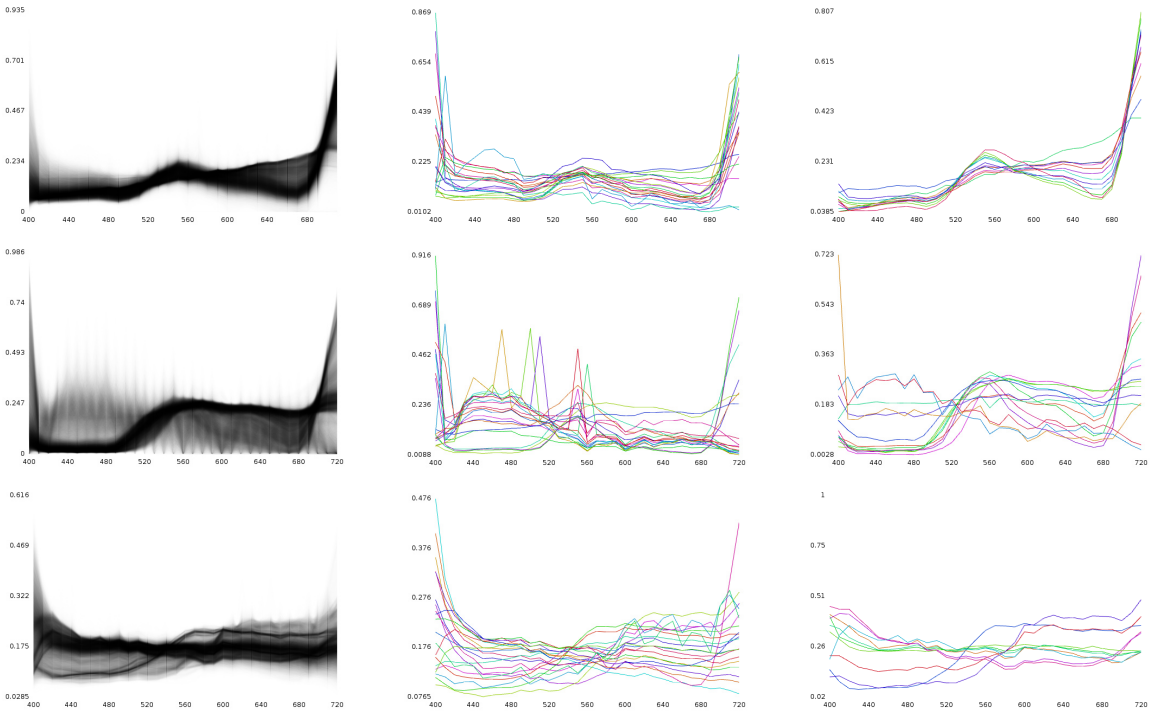
(a) Input in true color **(b)** HRK [Huyn 10b] **(c)** `mssom`

Figure 4.26: Material segmentation results on the *Cbrufefields*, *Cyflower*, and *Ribeira* scenes from the *Foster* dataset.

provides a usable clustering of the high-dimensional image data. Like with FSPMS, when providing a quantized feature space to FAMS, as is done with `mssom`, the characteristics of the segmentation can change for better or worse. A larger map is more robust in this case. It stays closer to the SG-FAMS result and we also found less segmentation errors in visual inspection of the results. The results of `somtopo` match SG-FAMS results more closely than one might expect, given the algorithmic differences. It appears most viable for clustering in an interactive setting, due to it being more reliable with a smaller SOM and as a result of that, its outstanding computational performance.

Material Prototype Learning

In this study, we investigate the particular problem of finding prototypes for materials contained in a natural scene [Jord 14]. We compare our SOM-based mean shift method, `mssom`, with the method by Huynh and Robles-Kelly, which we denote as HRK. A valid clustering and respective cluster modes, or means where modes are not determined by the algorithm, are supposed to be found on images from the *Foster* dataset. The dataset is well suited for benchmarking as it contains natural scenes with a high spatial resolution. As HRK employ the L_2-normalized data descriptor, we operate in this feature space for all methods in the comparison. We choose `mssom`



(a) normalized spectra **(b)** HRK [Huyn 10b] **(c)** `mssom`

Figure 4.27: Comparison of extracted prototypes for the *Cbrufefields*, *Cyflower*, and *Ribeira* scenes. (a) Normalized spectral distribution; (b), (c) Material prototypes yielded by HRK and `mssom`. Please refer to page 52 for an explanation of spectral distribution plots.

over `somtopo` due to its advantage of finding the mode in a cluster, which resembles the corresponding material prototype more likely than the mean of a segment.

The experimental results of [Huyn 10b] presented here were produced by Cong Phuoc Huynh for this comparison. Due to memory constraints, he performed a 1:2 subsampling of the images (a 4-fold reduction of image size without smoothing). HRK requires a user-determined number of clusters which was set to 20 in this experiment. For `mssom`, we train a 3-D SOM of size $n_M = 1000$. The mean shift step is run with $p = 0.6$ for the adaptive bandwidth selection. Both methods were run with the same parameter settings on all images.

Figure 4.26 shows segmentation results obtained by both methods on three images. While `mssom` found 13, 12, and 10 modes in the three images, respectively, the number of clusters was set to 20 for HRK to reduce the effect of under-segmentation. We observe that both methods are capable of discerning most relevant materials in the scene. Nonetheless, in the middle row of Figure 4.26b, we see that HRK, even with a higher number of clusters, suffers from under-segmentation over several image regions. At the same time, several clusters are generated from the noisy background of the image. In the third image, *Ribeira*, HRK achieves a good material separation of the scene, but still is affected by noise on the roof of the buildings, which cause cluster fragmentation.

Figure 4.27 depicts the material prototypes found in the respective images from Figure 4.26a. To derive these prototypes, in the case of HRK, the mean of each

cluster is considered as described in [Huyn 10b]. In the case of `mssom`, we take advantage of the mode-seeking and the prototypes correspond to cluster modes. The parallel coordinates plots of normalized spectral distributions in Figure 4.27a help to visually assess the material prototypes found by HRK and `mssom`. We can observe that both methods succeed in capturing the prominent materials in the scene. In the case of HRK, some of the clusters found are more descriptive than others. A good example is the *Cyflower* scene, where several prototypes delivered by HRK correspond to noise in the data. This produces "spiky" spectra that do not follow the intuition of smooth radiance in the spectral domain across the image.

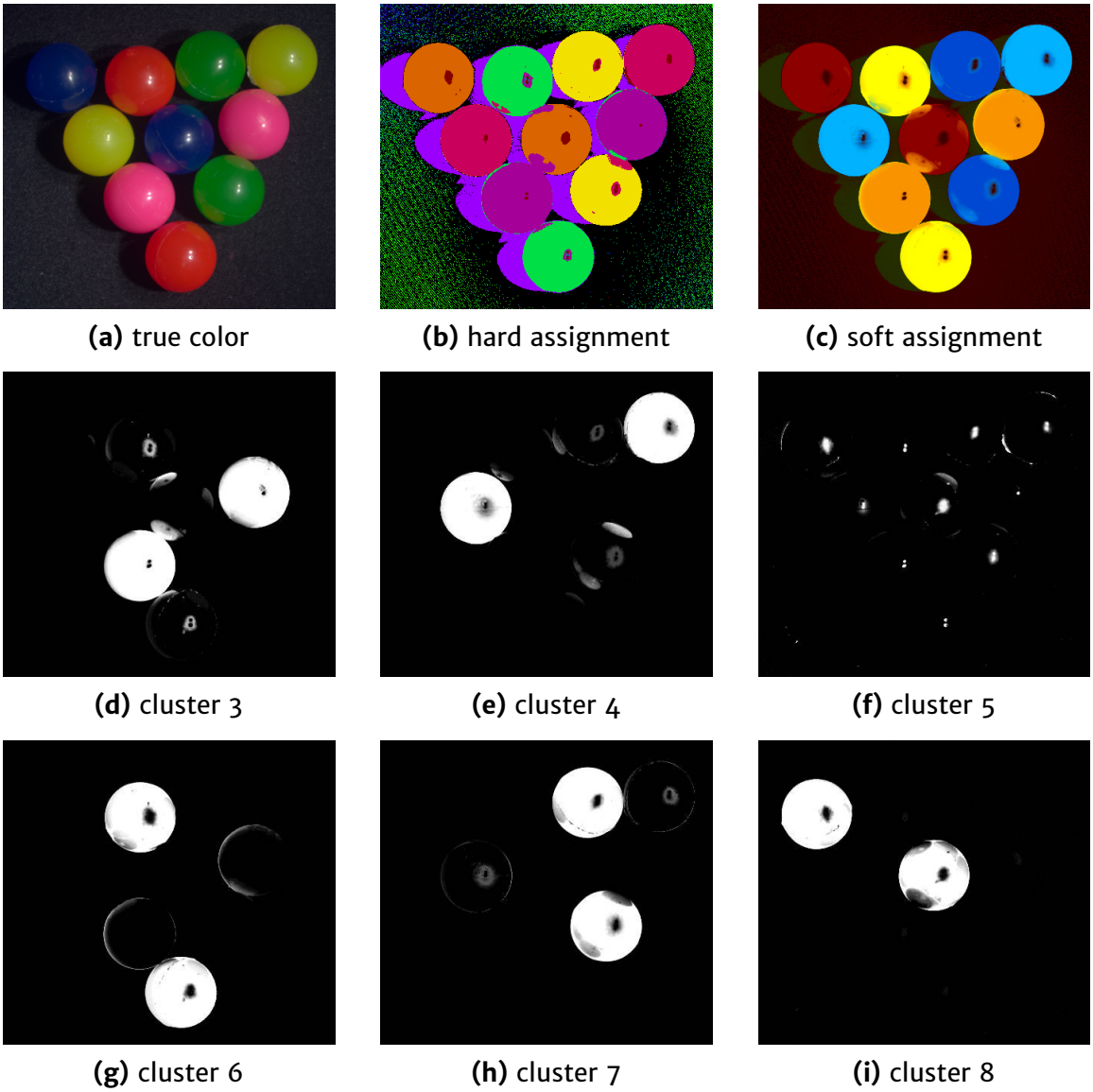
Per the motivation of our algorithm, results similar to `mssom` can be obtained with the baseline FAMS algorithm. An example is shown in Figure D.8a on page 178. Running our parallelized FAMS implementation on these images results in an execution time of around 10 to 11 hours per image from this dataset using an Intel Core i7-2600 CPU. The method of Huynh and Robles-Kelly took in average about six minutes when performed on a subsampled version of the images as discussed above. Execution times of `mssom`, however, stay well below a minute, faster than the results reported in [Jord 14] due to our optimizations as explained in Section 3.2.1.

Fuzzy Clustering

As noted in Section 4.2.3, the two-stage approach combined with a ranked BMU lookup allows for a fuzzy clustering, where each pixel is seen as a member of several clusters. To illustrate the use of such a clustering in multispectral image analysis, we test it on the *Superballs* image, which contains an array of plastic balls of various highly-reflective colors. The balls in the image are subject to several inter-reflections, some of which are hardly to be seen in the true-color display.

We run `mssom` on the spectral gradient of the image, with 20 Gaussian-weighted BMUs to form cluster memberships according to Eq. 4.35. In Figure 4.28, we depict the results of hard assignment next to a visualization of soft assignment and the pixel-wise weighting of individual clusters. The result in Figure 4.28f is interesting by itself, as it effectively provides an index of specular reflectance for each pixel. Note, by close inspection, how this includes specular highlights that were reflected from other balls. Furthermore, Figure 4.28d and Figure 4.28e reveal certain inter-reflections: Cluster 3 contains both the pink balls, and reflectances from pink or red on blue. Likewise, cluster 4 contains both the yellow balls, and reflectances from yellow on red, as well as red and pink on green. The cluster assignment of the inter-reflectances depends on colors of both the originating and the receiving material, reflecting the respective mixture. Finally, the clusters depicted in Figures 4.28g, 4.28h, 4.28i also show the regions of the balls in said clusters where either specular highlights or inter-reflectances occur and how strongly these negatively affect the mixture component of the respective clusters.

Our experiments reveal that unsupervised clustering of a multispectral scene is applicable and able to provide helpful results for scene analysis. Selection of an appropriate feature space for the application is crucial for effective clustering. Furthermore, both `mssom` and `somtopo` are fast, reliable and versatile methods that can be used in an interactive setting. Visual assessment of the clustering is



(a) true color **(b)** hard assignment **(c)** soft assignment

(d) cluster 3 **(e)** cluster 4 **(f)** cluster 5

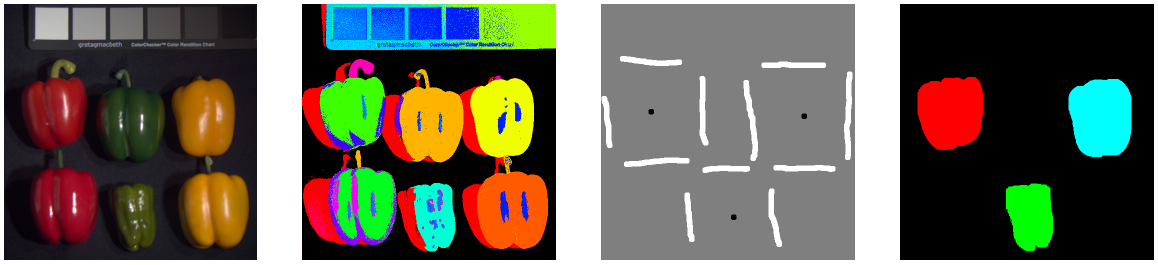
(g) cluster 6 **(h)** cluster 7 **(i)** cluster 8

Figure 4.28: Fuzzy clustering of *Superballs* image. The image is cropped for better visibility.

then possible through the distribution plots discussed in more detail in Section 5.3. We will now investigate the case where not a global clustering is desired, but the segmentation of specific objects in a scene, which may contain spectra originating from several clusters.

4.3 Supervised Segmentation

In an interactive analysis framework, input from the user can be a valuable prior to segmentation tasks. As a user explores the multispectral data step-by-step, they may want to compare the spectra of specific objects in the scene or examine reflectance properties of a certain area in detail. Automated segmentation replaces tedious manual labeling of this area. In Figure 4.29 we show the results of an unsupervised



(a) image in true color **(b)** SG-FAMS segmentation **(c)** User-provided seed points **(d)** supervised segmentation

Figure 4.29: Illustration of the difference in output between unsupervised clustering and seed-based segmentation on *Fake and Real Peppers* image.

segmentation on a test image using the SG-FAMS algorithm detailed in the previous section (Figure 4.29b) next to those of a guided segmentation (Figure 4.29d) that takes a manually created seed map (Figure 4.29c) as an additional input. The example illustrates that in certain scenarios, the ability to guide the algorithm lets the user obtain a segmentation that is much more helpful for a specific analysis task.

The concept of seed-based segmentation is based on incorporating user-provided prior knowledge of the scene and the task at hands in the form of image annotations. In the binary scenario, a foreground object is to be separated from the background. The user input can range from a loose trace of an object contour or a bounding box to a set of marked foreground and background pixels. Based on this information, all other pixels are then determined as belonging to either the background or the foreground. This concept can also be extended to multiple classes. We make it a powerful tool within our interactive workflow that will be described in Chapter 5.

4.3.1 Related Work

In the last decade, algorithms based on graph theory have dominated the field of supervised segmentation on both monochromatic and RGB images, after several powerful graph-based image segmentation algorithms were introduced. In 1997, Shi and Malik proposed the unsupervised normalized cuts algorithm [Shi 97]. In 2006, Leo Grady introduced random walks to image segmentation [Grad 06]. Both create a graph with a node for each pixel and edges between pixels in a local spatial proximity, whereas the edge weights are based on pixel dissimilarity. While the normalized cuts try to find a minimum cut of the graph that separates foreground and background, Grady's method computes the likely destinations of a random walker. A third paradigm to graph-based image segmentation are watersheds, where the intensity image is considered as a topographic relief [Cous 09].

In 2009, Couprie et al. introduced a framework for supervised segmentation that incorporates several key methods based on graph theory [Coup 09, Coup 11]. Their *power watersheds* integrate graph cuts [Boyk 06] with the aforementioned random walker and watersheds in a single mathematical framework.

For this algorithm family, the input consists of three parts. First, the pixels $\mathcal{X}$, which are strictly used in a differential manner. Second, the spatial location of each pixel. Third, two sets of pixel locations, the foreground seeds $\mathcal{F}$ and background

seeds $\mathcal{B}$. The spatial relation of pixels is reflected in a graph structure. Each pixel $\boldsymbol{x}_i$ corresponds to a vertex $v_i \in \mathcal{V}$. The edge set $\mathcal{E}$ connects vertices in a 4-connected lattice. The edge weight w_{ij} of an edge e_{ij} is a function of the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ and will be examined in more detail later.

We compute an n_X-element vector $\boldsymbol{p}$, where p_i is the probability for $\boldsymbol{x}_i$ to belong to the foreground or background class via

$$\begin{aligned} \boldsymbol{p}^* = \underset{\boldsymbol{p}}{\operatorname{argmin}} \sum_{e_{ij}\in\mathcal{E}} w_{ij}^p \left|p_i - p_j\right|^q + \sum_{v_i\in\mathcal{V}} w_{Fi}^p |p_i|^q + \sum_{v_i\in\mathcal{V}} w_{Bi}^p \left|p_i - 1\right|^q \ , \\ \text{s. t. } \forall i \in \mathcal{F} : p_i = 1, \forall i \in \mathcal{B} : \ p_i = 0 \ , \end{aligned} \tag{4.37}$$

where w_{Fi} and w_{Bi} denote unary weights penalizing foreground and background affinity. Simple thresholding leads to a binary segmentation s with $s_i = 1$ (foreground pixel) if $p_i \geq \frac{1}{2}$, and 0 (background pixel) otherwise. Based on the selection of parameters p and q, this minimization matches the graph cuts (p positive finite, $q = 1$), random walker (p positive finite, $q = 2$), or shortest paths (p and q converge toward infinity together with the same speed) algorithms. With $p = \infty, q \geq 1$, the power watershed algorithm is obtained [Coup 11].

Established applications of this method are 2-D intensity images, RGB images, as well as 3-D medical data. For these, Couprie et al. define the edge weights

$$w_{ij} = \exp\left(-\beta(\mathrm{d}(\boldsymbol{x}_i, \boldsymbol{x}_j))^2\right) \ , \tag{4.38}$$

where β is a constant and $\mathrm{d}(\cdot,\cdot)$ is the L_2 norm [Coup 09] or the L_∞ norm [Coup 11]. For hyperspectral data, these choices are reasonable if a single band or the PCA of the image is used as input. However when operating on the full spectra of the image, a more appropriate distance function should be employed.

4.3.2 Hyperspectral Power Watersheds

We identified the edge weighting as a primary concern when adapting the Power watershed algorithm to hyperspectral data. As discussed in Section 2.4, L_p-based distances are often limited in comparing high-dimensional spectral vectors. Therefore, we evaluated a range of distance functions on multispectral data. This work was first described in [Jord 12b].

Since a common goal of supervised segmentation is to discriminate specific objects in a scene, a similarity measure that discerns materials is a fitting choice for $\mathrm{d}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. For this application, we examined the measures SA, NED, SID, SIDSAM_1, and SIDSAM_2, as defined in Section 2.4.2 (see pages 21–23). As the difference in the results of SIDSAM_1 and SIDSAM_2 is negligible, we only report results for SIDSAM_1.

Power Watersheds with $q = 2$ is effectively a graph-cut algorithm where a random walker is employed in the special case of plateaus. A plateau describes a subgraph where all edge weights are equal, and the random walker works as a regularizer to provide a smooth, reasonable cut through the plateau. Without the random walker, the cut could happen on arbitrary edges, e.g. resulting in a zig-zag segment border. In the original implementation for 8-bit monochromatic or color-triplet data, the plateau detection could simply test for equal edge weights (stemming directly from

intensity differences, as seen in Eq. 4.38), while in our case of non-integer weights, a simple test for equality fails. We can solve this problem by employing a bucket-sort algorithm [Esti 99]. However, highly non-uniformly distributed dissimilarities e. g. in the case of SAM and SID, do not work well with a fixed bucket size. To avoid this problem, we can apply histogram-equalization on the edge weights as explained in Section 2.4.2 (page 23). Note that this has neither an effect on the graph-cut or the random walker algorithms separately. The former is only taking the order of edge weights is taken into account, but not their ratio, while the latter is confined to the plateau.

Data-driven measure

In Section 4.1, we showed that by using the topological distance inside a self-organizing map, we obtain valuable differential information between adjoining pixels. For this, the data-driven pseudometric SOM_1 is given in Eq. 4.9 as

$$\mathrm{SOM}_1(\boldsymbol{x}, \boldsymbol{y}) = \left\| \boldsymbol{r}^{\left(c^{(x)}\right)} - \boldsymbol{r}^{\left(c^{(y)}\right)} \right\|_2 , \tag{4.39}$$

where $\boldsymbol{r}^{\left(c^{(x)}\right)}$ is the location in the map representing $\boldsymbol{x}$. As this measure is well-suited for edge detection, it is a natural extension to also use it in the context of graph-cut based segmentation methods. The particular strengths of a SOM-based similarity measure are its flexibility and its generalizability, as it adapts itself to the present data.

Figure 4.32 on page 116 shows, for the tested similarity measures, example gradient maps in the x-direction by computing the similarity of each pixel with the pixel to its right. Edge weights in a 4-connected lattice are computed from this data and the gradient in y-direction resp. to Eq. 4.38.

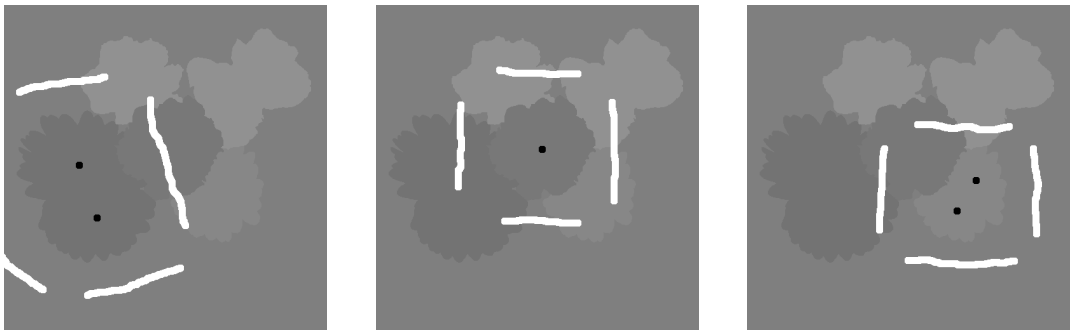
4.3.3 Experimental Results

To the best of our knowledge, no ground-truth segmentation is available for any of the publicly available multispectral or hyperspectral datasets. To close this gap, we define a comprehensive set of segmentation tasks with associated ground-truth segmentations on the CAVE multispectral image dataset [Yasu 10]. See Table 2.2 on page 12 for the characteristics of this dataset.

Segmentation Tasks

We chose this dataset as it is well-suited for generating ground-truth segmentations, consisting of high-quality multispectral images that depict objects of different materials in a laboratory setting. We hand-labeled certain objects in nine images that cover a good variety in difficulties. Labeling was performed within our software framework, using the visualization capabilities described in Chapter 5. The hand-labeling was done by carefully considering all spectral gradient bands of an image in which the object under investigation is clearly separated from its surroundings. In total we have 32 segmentation tasks.

(a) ground truth **(b)** task 1 **(c)** task 2



(d) task 3 **(e)** task 4 **(f)** task 5

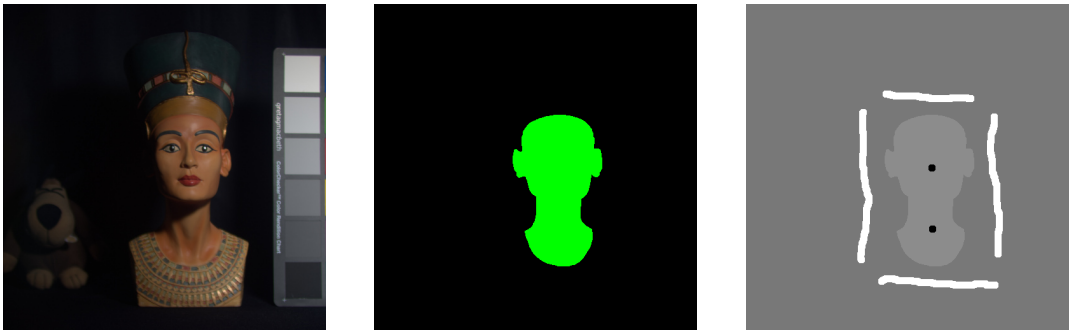
Figure 4.30: Segmentation tasks for *Flowers* image. For each task, one flower is defined as foreground, the rest of the image as background. The image is depicted in Figure 2.2 on page 7.

We also provide seed point input for the specific tasks: We place foreground seeds in the form of a circle with a 5-pixel radius in the center of each object. Background seeds are placed as hand-drawn lines on the top, left, bottom and right of each object with a distance of 20 to 40 pixels to the object contour. The seeds mimic a typical usage scenario.

Figure 4.30 shows ground-truth labels and seed inputs for the five tasks defined for the *Flowers* image. Note that the algorithm only sees a ternary map: Black foreground seeds, white background seeds and gray undetermined pixels. See Figure 4.31c, Figure 4.33b, and Figure 4.29c (three tasks combined) for further examples. We provide these segmentation tasks freely accessible on the web [Jord 12a].

Benchmark

Our benchmark is two-fold: On the one hand, we compare the different edge weightings L_∞, SA, SID, SIDSAM, NED and SOM. On the other hand, we test four different algorithms that are included in the framework of Couprie et al. First is the graph cut based on a maximum spanning forest computation, `mfs`. Second is the power watershed algorithm with $q = 2$, `pw`, that includes a random walker. The other two algorithms are based on these, however they employ geodesic reconstruction, `msfg` and `pwg`, respectively. Parameters of these algorithms are kept to the implementation defaults of [Coup 11], except for the new edge weighting. For the SOM similarity,



(a) image in true-color **(b)** ground truth **(c)** seed points

Figure 4.31: Seed input for *Egyptian Statue* image used to generate the results shown in Figure 4.32.

a 32×32 SOM is trained. Training takes less than 5 seconds and is included in reported execution times.

From the difference between ground-truth and segmentation result we compute precision, p, (the probability that a machine-generated foreground pixel is a true foreground pixel), recall, r, (the probability that a true foreground pixel is detected) and F_1-score,

$$F_1 = 2\frac{p \cdot r}{p + r}\,. \tag{4.40}$$

These quantities are suitable for our task as they do not depend on image size, but only the number of machine-generated foreground pixels and true foreground pixels. We use a color-coding when displaying detection performance on individual image and task combinations as follows. *White:* detected true foreground pixel; *Dark green:* missed true foreground pixel; *Red:* Background pixel misclassified as foreground pixel. *Black:* background pixel, correctly classified.

Results

In Table 4.3 the average performance over all images is reported. Average execution time (in seconds) includes training once per image for SOM. As the geodesic reconstruction had little to no effects even on the single results, `mfsg` is omitted. In Table 4.4, results per-image are presented for each similarity measure in its overall best performing algorithm combination.

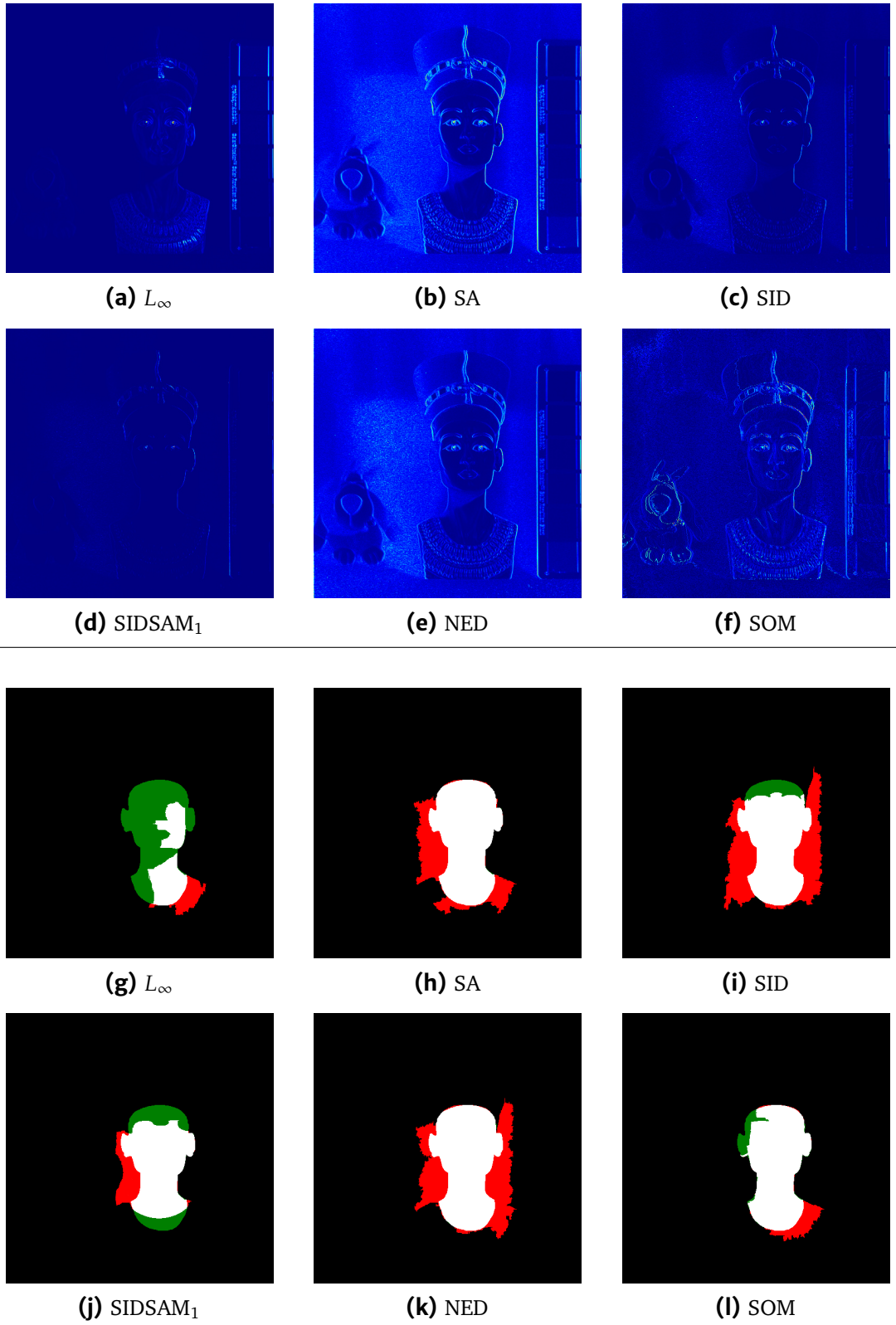
It can be observed that the power watershed algorithm performs well for our application. The use of an edge weighting specific to our domain is important. This is indicated by the bad performance of L_∞, which is a good choice for RGB images. The most successful similarity measures SA and NED perform similarly, both qualitatively and quantitatively. They share the property of scale invariance, but are prone to noise in dark image regions. This explains their considerably bad performance on the *Egyptian Statue* image shown in Figure 4.32 (algorithmic input shown in Figure 4.31a, 4.31c). Here, the foreground object is not well separated from the background due to being partly self-shadowed. SA (see Figure 4.32b) and NED respond more strongly to noise in the background than to the object boundary

Measure	Algorithm	Precision	Recall	F_1-score	Time (s)
NED	pwg	0.908	0.971	**0.929** ± 0.073	6.0
SA	pw	0.896	0.978	**0.928** ± 0.067	6.8
SA	msf	0.892	0.978	**0.926** ± 0.065	0.7
SA	pwg	0.891	0.979	**0.923** ± 0.071	6.8
NED	pw	0.911	0.962	**0.919** ± 0.089	1.0
NED	msf	0.907	0.962	**0.917** ± 0.091	1.0
SOM	pwg	0.897	0.925	**0.898** ± 0.074	18.6
SOM	pw	0.897	0.925	**0.898** ± 0.074	18.6
SOM	msf	0.894	0.903	**0.886** ± 0.057	9.6
SID	msf	0.846	0.973	**0.867** ± 0.249	2.8
SIDSAM	msf	0.819	0.972	**0.849** ± 0.245	3.4
SID	pwg	0.888	0.872	**0.837** ± 0.210	8.5
SID	pw	0.889	0.872	**0.837** ± 0.210	8.6
SIDSAM	pw	0.944	0.764	**0.793** ± 0.259	10.9
SIDSAM	pwg	0.947	0.763	**0.791** ± 0.266	10.7
L_∞	pw	0.937	0.588	**0.659** ± 0.214	8.9
L_∞	pwg	0.923	0.587	**0.655** ± 0.217	8.9
L_∞	msf	0.927	0.538	**0.598** ± 0.291	0.9

Table 4.3: Average performance of measure/algorithm combinations.

	Balloons / 4	*Statue* / 1	*Food* / 4	*Lemons* / 2	*Peppers* / 3
L_∞	0.97 ± .04	0.51	0.72 ± .24	0.46 ± .40	0.50 ± .35
SA	0.97 ± .03	0.84	**0.99** ± .01	**0.99** ± .01	**0.99** ± .01
SID	0.96 ± .03	0.21	0.98 ± .01	**0.99** ± .00	**0.99** ± .01
SIDSAM	0.96 ± .03	0.21	0.98 ± .01	0.95 ± .05	**0.99** ± .01
NED	0.97 ± .03	0.77	0.98 ± .01	**0.99** ± .00	**0.99** ± .01
SOM	**0.98** ± .01	**0.91**	0.94 ± .05	0.98 ± .00	0.95 ± .05
	Feathers / 6	*Flowers* / 5	*Toys* / 2	*Superballs* / 5	
L_∞	0.82 ± .09	0.50 ± .29	0.48 ± .04	**0.97** ± .02	
SA	0.83 ± .17	**0.94** ± .04	**0.94** ± .05	0.86 ± .17	
SID	**0.89** ± .06	0.90 ± .13	0.92 ± .02	0.96 ± .05	
SIDSAM	0.82 ± .17	0.89 ± .13	**0.94** ± .04	0.90 ± .12	
NED	**0.89** ± .07	0.87 ± .15	**0.94** ± .05	0.96 ± .03	
SOM	0.83 ± .17	0.89 ± .07	0.77 ± .03	0.83 ± .13	

Table 4.4: Average F_1-scores per edge weighting method on each image. The number of segmentation tasks is denoted next to the image name.



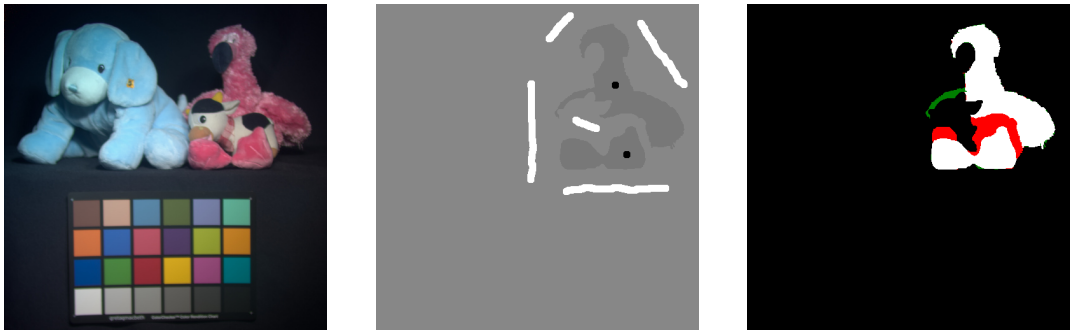
(a) L_∞ **(b)** SA **(c)** SID

(d) SIDSAM_1 **(e)** NED **(f)** SOM

(g) L_∞ **(h)** SA **(i)** SID

(j) SIDSAM_1 **(k)** NED **(l)** SOM

Figure 4.32: Gradients according to different similarity measures in x-direction (top) and their segmentation results (bottom) on *Egyptian Statue* image.



(a) Image in true color **(b)** Seed points **(c)** SA

(d) NED **(e)** SIDSAM **(f)** SOM

Figure 4.33: Example segmentation results on *Stuffed Toys* image, one of two tasks.

in the shadowed region. This is a case where the SOM similarity measure draws an advantage from being trained on the specific image.

The SID measure and its combinations with SA, $\text{SIDSAM}_{1,2}$, do not perform as well in our experiment. While results are often on par with SA, they are not reliable, as revealed by the high standard error. The measure poses specific problems to power watersheds, which makes the less powerful graph cut algorithm perform better. The SOM-based similarity did not perform best in this experiment. However, the results show that the SOM did indeed adapt well to the application, purely based on the data presented to it in training, while the other tested similarity measures where specifically designed for material discrimination based on spectrum. The SOM can be further adapted to a specific scenario by changing parameters of training and read-out, while the other measures are fixed. Typically, the right measure needs to be chosen based on application domain [Guti 10], while the SOM approach is general by design.

Figure 4.33 depicts one of the most challenging tasks in the benchmark. The foreground object is partially occluded. Here SIDSAM performs best. SA and NED produce very similar results. The complete segmentation results are available at [Jord 12a].

Our experiments reveal that algorithms from the power watershed family can deliver a strong performance for multispectral segmentation. A straightforward attempt of using the Chebyshev distance does not yield satisfactory results. However by using other similarity measures, the method is reliable even when only few foreground seeds are placed and background seeds only provide a rough, disconnected

outline of the object. The graph-based segmentation method is remarkably robust against imbalanced edge weights, disconnected boundaries and noisy regions, where single pixels elicit random, strong edge weights with their neighbors. This is why, in some cases, similarity measures that performed poorly in edge detection significantly gain in performance when combined with this algorithm. Nevertheless, we observe that in some cases, the SOM_1 measure outperforms other measures significantly. An investigation in how SOM_2 would improve on SOM_1 for this task is subject to future work.

4.4 Discussion

In this chapter, we visited three important fields in image processing for the analysis of multispectral and hyperspectral images.

We first addressed the difficult problem of edge detection on images with many bands. Most work on edge detection has been done on grayscale images, with extensions to tristimulus data. Exceptions are a method by Toivanen et al., based on a global ordering and prone to producing artifacts, and SEDMI, based on ensemble clustering, which is slow to compute. We found that the high dimensionality of hyperspectral data is a challenge for other methods, which are either based on gradients or vector ordering. These approaches highly dependent on a suitable similarity measure. Existing measures, even when designed specifically for hyperspectral data, fall short in some of our test images. We proposed two variants of a data-driven pseudometric that is based on a self-organizing map. The second variant improves results by a changed SOM lookup algorithm. Both pseudometrics are extended to mimic a Sobel filter, which comes handy when applying them with the popular Canny edge detector. In our experimental evaluation, we found that while the vector-ordering based RCMG algorithm improves on edge detection results of the Laplacian method, it still suffers from deficiencies of the established similarity measures. The proposed measures deliver good results on our test images for both the Laplacian and RCMG edge detectors. They also perform favorably when compared with SEDMI for both continuous edge maps (RCMG) and binary edge maps (Canny).

Second, we investigated global segmentation, or clustering, of the image. Other than what is common for regular color images, the data is clustered in the hyperspectral feature space without spatial context. However, we found that the use of a derived data descriptor, e. g. L_2-normalized or spectral gradient, can be of significant help for material-based clustering. Methods exist that can produce satisfactory segmentations of multispectral and hyperspectral data. One prominent example is the mean shift algorithm. However, it is slow to compute even for images of small dimensions. We derived two methods for speedup for the FAMS algorithm, which is a popular variant of high-dimensional mean shift. One works with an adaptive reduction in spatial resolution. The other is based on a quantization of the feature space derived by a SOM. Both methods allow clustering in interactive time constraints. When reviewing work on SOM-based clustering, we realized that as of now, the learned topology is not used most effectively. Building on recent work in the field of superpixel segmentation, we derived the `somtopo` algorithm as a new topology-based SOM clustering method. When compared to the mean

shift variants, it has the advantage of a simpler design: It does not need bandwidth selection or mode pruning steps, and is overall faster to compute. The experimental results reveal that obtained segmentations are generally comparable to the output of the baseline FAMS algorithm. The `somtopo` algorithm can be recommended for an interactive setting, where fast computation is of utmost importance. During an analysis session, the user can easily fine-tune segmentation parameters as well as results.

We then moved to a scenario where the user defines a specific segmentation goal by providing a set of points to be included in a segment or left out, called seeds. Therefore, spatial content is very relevant to the segmentation, which advises the use of a graph-based segmentation algorithm. We built our work on the mathematical framework proposed by Couprie et al.. It includes the most prominent seed-based methods as well as an extension under the name power watersheds. To the best of our knowledge, no prior work exists on seed-based segmentation of multispectral or hyperspectral images. We designed a benchmark for such algorithms based on the publicly-available *CAVE* image dataset. The tasks in the benchmark mimic a typical usage scenario, however seeds are placed purposefully scarce. Although the tasks are generally challenging, the proposed method already produces convincing results. In practice, users tend to provide richer seed information, which makes the functionality even more reliable for day-to-day use.

The work carried out provides capable tools for interactive inspection of hyperspectral data. In Chapter 5, we discuss visualization methods that match the automated analysis to form our software framework.

Chapter 5

Visualization

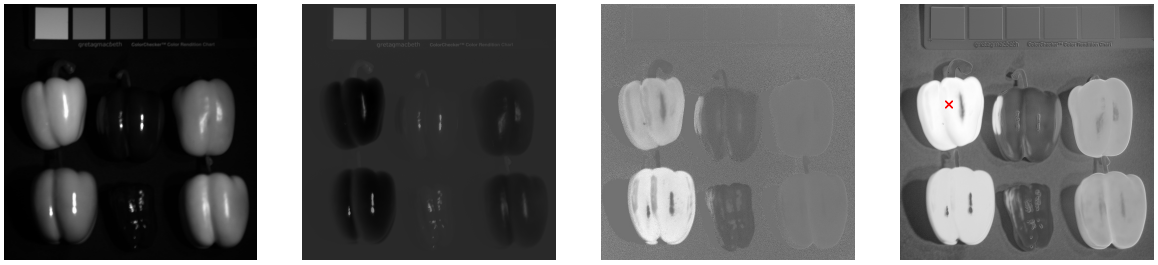
The high dimensionality of hyperspectral data makes it inaccessible to the human observer at first. Likewise, it is difficult to analyze high-dimensional data, or find relevant information in it, by automated processing alone. Unsupervised algorithms may fail to overcome the challenges that come with multispectral and hyperspectral images. Qualitative analysis is necessary to assess the plausibility of algorithmic results, but it can also be helpful for generating prior knowledge. Consider an image taken for the first time for a specific application, e. g. exposing hidden or obscured text on a palimpsest. With the right tools, we can discover a great amount of information about said image through manual analysis that helps us in designing an algorithmic solution to our problem. For example, we might be able to deduce the most relevant bands for our application or how many different materials can be found in the spectral data.

In this chapter, we first present common approaches to visualization of multispectral and hyperspectral imagery. Traditional methods include data displays within the spatial layout of the image or commonly-used plotting tools. A popular visualization technique coined *false coloring* assigns a color to each pixel that encodes most-relevant data related to the pixel. We present a new false coloring technique that is learned in an unsupervised manner and is designed to quickly expose disparities in the data.

We then move to interactive visualization, as opposed to the currently used static displays. Interactivity allows us to present the data in a very rich fashion that is complimentary to existing visualizations. It gives direct visual access to the spectral distribution of the whole image, or specific objects in the scene, and their relation to each other. This is even more powerful when selecting or combining different data descriptors. As a matter of fact, certain spectra of interest can often be identified by simple intensity thresholding in a few bands of a suitable descriptor. Our interactive visualization can reveal this information to the user in short time.

5.1 Related Work

The simplest displays for a hyperspectral image are the grayscale representation of single bands and the plotting of spectra. The ‘spectral lens’ by Kim et al. allows to



(a) image band at 680 nm **(b)** second PCA component **(c)** spectral gradient at 585 nm **(d)** similarity map

Figure 5.1: Various displays of *Fake and Real Peppers* image shown in Figure 4.29a and Figure 5.6a. The similarity map was computed via the spectral angle between each pixel in question and the pixel marked in red.

show a region of one band in the spatial context of another band [Kim 10]. Spectra are often plotted for a small selection of pixels or a selected slice (a path in the image, e. g. a linescan). Another possibility is to plot the mean and variance in each band for a specific image region, for example to compare different labeled areas. Histograms are also employed in this regard. Interband correlation, as seen in Figure 2.11 on page 25, may be plotted for a labeled area to provide the statistical variation present in a class [Bieh 02].

More sophisticated displays include scatter plots or biplots that relate selected regions in a pair of bands [Exel 16, Bieh 02]. A similarity map may reveal related regions in an image by displaying a heatmap of similarity measured between all pixels and a selected pixel, or the mean vector of a selected region. Figure 5.1 depicts such a map next to single bands from various image descriptors selected to best discriminate the pepper on the top-left. The use of 3-D rendering is generally limited to the domain of astronomy [Joye 03, Li 08], where the vast majority of the scene consists of negligible background that may be removed by simple thresholding, so that a clutter-free presentation of foreground objects is possible. Other visualizations exist that exploit additional knowledge available for an image. Labitzke et al. use labeled segments for Radviz-based visual analysis [Labi 13b], while Cai et al. display an array of pie charts based on known abundances [Cai 07].

A popular and useful method of visualizing multispectral and hyperspectral data is the generation of a color image that preserves the spatial relations in the image, but encodes specific information in the coloring [Biou 13]. We now consider in more detail how such color images are obtained.

5.1.1 Color mapping

While a grayscale image can present a single piece of information per-pixel, a color image displays three data values within the `r`, `g`, `b` color triplet. There are four common categories of color displays for hyperspectral image data.

Band composite Three image bands are selected and then mapped to `r`, `g`, `b` color channels. This is a very common technique, however the three bands in question are typically hand-selected. They might also be obtained by statistical measures, e.g.

(a) band composite **(b)** true color **(c)** PCA false color

Figure 5.2: Color visualizations of *Indian Pines* image. To generate (a), the bands at 841 nm, 637 nm, and 538 nm were selected for the red, green, and blue channels, respectively.

the entropy in each band or correlation between selected bands [Bajc 04, Tsag 05], or the structure in a band [Demi 09]. More work on general band selection was discussed in Section 3.1.1 on page 29.

Data fusion A subset of bands is fused according to a specific criterion. Most often, this is the human color perception, modeled by the CIE XYZ color-matching functions [Wysz 00, Chapter 3] and referred to as *true color*. This method is explained in more detail in Section 5.1.2. Jacobson et al. derive from this concept by offering different basis functions [Jaco 07]. Kotwal and Chaudhuri compute a weighted average of all bands with data-dependent weights [Kotw 10].

Dimensionality reduction The entire spectral information is reduced to three dimensions, which are mapped to `r`, `g`, `b`. Ideally, the information most relevant to the application in question is covered in these three dimensions. We discuss dimensionality reduction methods in Sections 5.1.3 and 5.2. Tsagaris et al. also suggest to perform band selection before this step [Tsag 05].

Feature extraction A combination of application-specific indicators (e.g. abundances of three user-selected endmembers) forms the display. For example, it may depict the occurrence of forest types to assess forest structure in [Usti 00]. It is common to visualize indicators like the vegetation index in pseudo-color displays [Habo 04]. As these methods are typically application-dependent, they do not fit our goals of providing a general approach to visualization.

As they are the most commonly found displays that can be computed in an unsupervised fashion, we examine color mapping based on both human color perception, as well as based on dimensionality reduction more closely. Figure 5.2 illustrates common color visualizations on a remote sensing image.

5.1.2 True Color

For images whose spectrum lies within the range of visible light, an intuitive natural representation can be formed by mimicking human color perception, based on experimentation carried out to measure the sensitivity of the human eye towards light of different wavelengths [Wysz 00, Chapter 5]. As the definition of *color* is

purely perceptional, this representation is often referred to as *true color*. Two types of photoreceptor cells exist in the human eye. Rod cells provide monochromatic vision, while Cone cells are responsible for color vision [Wysz 00, Chapter 2]. We have a trichromatic vision based on the three different types of Cone cells in our eyes, each with their respective response curves in relation to the wavelength of the perceived light. Therefore, all color sensations are based on a tristimulus which can be emulated.

The CIE 1931 colorimetric system defines three color-matching functions $\overline{x}(\lambda)$, $\overline{y}(\lambda)$, $\overline{z}(\lambda)$ [Wysz 00, Chapter 3]. These functions describe the spectral sensitivity of a human observer, called the CIE 1931 standard colorimetric observer. The tristimulus values X, Y, and Z are then obtained as

$$\begin{aligned} X &= k \int_{360\,\mathrm{nm}}^{830\,\mathrm{nm}} P_\lambda \overline{x}(\lambda) d\lambda \,, \\ Y &= k \int_{360\,\mathrm{nm}}^{830\,\mathrm{nm}} P_\lambda \overline{y}(\lambda) d\lambda \,, \\ Z &= k \int_{360\,\mathrm{nm}}^{830\,\mathrm{nm}} P_\lambda \overline{z}(\lambda) d\lambda \,, \end{aligned} \tag{5.1}$$

where P_λ denotes the intensity observed by a pixel $\boldsymbol{x}$ at wavelength λ and k is a constant factor that we set to normalize the output intensity. In practice, the integrations are replaced by summations over bands of equal width $\Delta\lambda$ centered at wavelength λ [Wysz 00, Chapter 3]. Popular filter-based multispectral sensors capture the incoming light with $\Delta\lambda = 10$. The X, Y, and Z values are directly used in the CIE XYZ color space. They can be transformed to RGB given a reference white and a gamma correction value. The sRGB space has default values for both and is specifically designed for computer displays [Stok 96]. To transform from XYZ to sRGB, we carry out the linear part,

$$(r, g, b)^{\mathrm{T}} = \begin{pmatrix} 3.2410 & -1.5374 & -0.4986 \\ -0.9692 & 1.8760 & 0.0416 \\ 0.0556 & -0.2040 & 1.0570 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} , \tag{5.2}$$

before applying the nonlinear gamma correction function

$$\gamma(x) = \begin{cases} 12.92x & x < 0.00304 \\ 1.055x^{\frac{1}{2.4}} - 0.055 & \text{otherwise} \end{cases} , \tag{5.3}$$

such that the red, green, and blue color channel of a pixel representation suitable for display are derived from r, g, b, respectively,

$$\mathrm{sRGB}(\boldsymbol{x}) = \big(\gamma(r), \gamma(g), \gamma(b)\big)^{\mathrm{T}} . \tag{5.4}$$

Artificial Illumination

In a true color display, for several applications, e. g. color restoration [Mori 08], analysis of paintings [Cola 06], and historical document analysis [Kim 10], scene

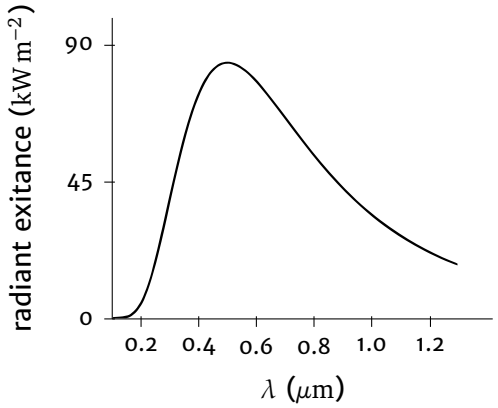


Figure 5.3: Blackbody radiator at $T = 5500\,\mathrm{K}$, which approximates vertical daylight.

illumination is an important characteristic. As hyperspectral images provide the complete spectral response of each pixel, illumination is easy to manipulate. In most data capture scenarios, homogeneous scene illumination can be assumed. In such cases, the illuminant spectrum can be removed from the data, either via direct calculation if the light source is explicitly known, or indirectly within a spectral normalization protocol that is applied to the raw sensor data. We can then easily manipulate the illumination curve.

According to Eq. 2.1 (see page 19) the observed intensity $\mathrm{I}(\boldsymbol{p}, \lambda)$ is formed by a combination of the illumination spectrum $\mathrm{e}(\lambda)$ and geometrical and material effects, which are independent of $\mathrm{e}(\lambda)$, $\mathrm{R}(\boldsymbol{p}, \lambda) = \mathrm{E}(\boldsymbol{p})\,\mathrm{S}(\boldsymbol{p}, \lambda)$. If the spectrum $\mathrm{e}(\lambda)$ of the incident light is known, the normalized reflectance of a pixel becomes: $\mathrm{R}(\boldsymbol{p}, \lambda) = \mathrm{I}(\boldsymbol{p}, \lambda)/\mathrm{e}(\lambda)$. Multispectral images are often already normalized with respect to illumination and sensor sensitivities, as discussed in Section 2.3.1. If so, we can assume $\mathrm{R}(\boldsymbol{p}, \lambda) = \mathrm{I}(\boldsymbol{p}, \lambda)$. In any case, a new illuminant spectrum $\mathrm{e}^*(\lambda)$ can then be applied by setting $\mathrm{I}^*(\boldsymbol{p}, \lambda) = \mathrm{R}(\boldsymbol{p}, \lambda) \cdot \mathrm{e}^*(\lambda)$. Due to the spectral sampling, the intensity of a certain band does not contain the information of a single wavelength but the integral over a range of wavelengths. Therefore, applying these calculations to the image bands is only an approximation. However, these approximations can be justified by the typically narrow filter bandwidth of multispectral sensors and the widely-accepted observation that the spectra of most surfaces and illuminations are smooth functions.

Several reference illuminants are available for image relighting. We model them as black body radiators [Wysz 00, Chapter 1]. Figure 5.3 depicts the radiant exitance curve of a blackbody radiator. Empirical measurements of the daylight spectra have shown that outdoor light, as well as indoor illuminants closely fit the spectra of black body radiators [Hend 63, Hend 64]. They, in turn, can be described by Planck's law, where the illuminant color is parametrized by the color temperature T in Kelvin. Daylight has a color temperature between 5000 K and 6500 K, while tungsten light bulbs have a temperature $T \approx 2800\,\mathrm{K}$. Figure 5.4 shows an example of a natural scene being set into two different lighting situations for the true-color display.

The application of different illuminants to the display may reveal the effect of metamerism [Fost 06, Wysz 00, Chapter 3]. Metamers are objects of varied

(a) daylight, $T = 6500\,\mathrm{K}$ **(b)** Tungsten, $T = 2880\,\mathrm{K}$

Figure 5.4: True-color displays of *Cyflower* image under two different artificial illuminants. The images were enhanced in brightness and contrast for display.

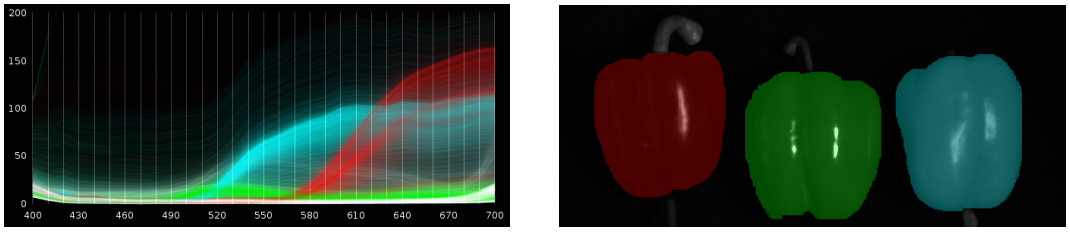
reflectance properties that yield the same color sensation. In general, however, this effect can render true-color representations and their variants unsuitable for several applications. This is where *false coloring* comes into play.

5.1.3 False Color

A *false-color* image neglects the natural color perception to put emphasis on different characteristics of the data. Very prominent in false coloring are PCA-based methods [Jaco 05]. We recall from Section 3.1.1 (page 28) that via the PCA we find a transform that maximizes the spread of the high-dimensional spectra over a small number of principal components. To find the principal components in the data, eigenvalues are computed. The eigenvectors corresponding to the three largest eigenvalues form a linear transformation to three bands used as `r`, `g`, `b` values of a color image. The component corresponding to the largest eigenvalue is thereby used for the green channel, where human color sensitivity is highest. Due to the nature of PCA, the per-component variance differs significantly. Therefore, an automatic white balancing is performed for display purposes [Jaco 05].

Tyo et al. propose a transform of the three principal components to HSV colorspace in an opponent-color fashion [Tyo 03]. Unfortunately, the method needs supervision for defining offsets to the second and third principal component. A notable derivation from PCA-based methods is described by Cui et al. and also operates in HSV color space [Cui 09]. To maximize spectral distance preservation, principle components form only `H`, `S` components. The `V` component is then calculated by optimizing a L_2-based distance similarity objective. However, as was seen in Section 4.1 and Section 4.3, the informative value of the Euclidean distance for spectral similarity is arguable. The same objective is used in a more general optimization method by Mignotte [Mign 12]. The independent component analysis (ICA, Section 3.1.1 on page 29) visualization is a linear transformation similar to PCA that seeks mutually independent components in the data [Wang 06b]. However, for a mapping to `r`, `g`, `b`, the most significant channels must be chosen. No rule is established for how to rank the significance of ICA channels [Cui 09].

Performing a linear transform and using three bases that explain most of the variance in the data is not always the most helpful representation for discerning



(a) spectral distribution **(b)** labeling

(c) true color **(d)** PCA false color

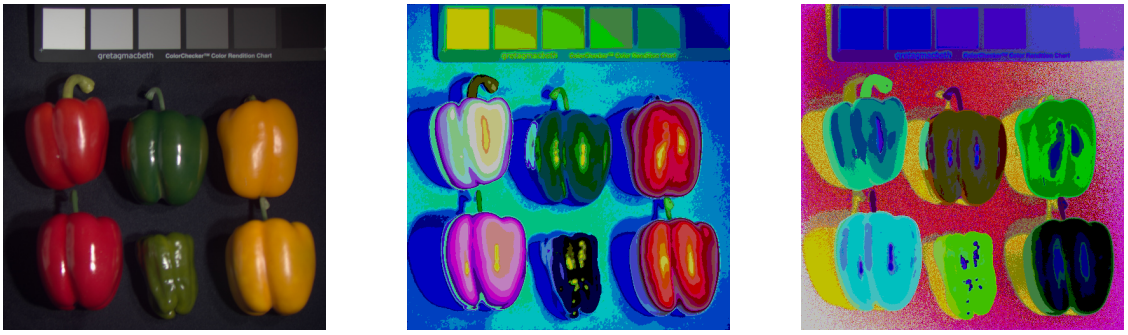
Figure 5.5: Displays of cropped *Fake and Real Peppers* image. The spectral distribution, (a), is label-colored. (b) shows the respective labeling as an overlay on the image band at 530 nm.

relevant spectra. Figure 5.5 shows a counter-example. The contrast of the green pepper against the background in Figure 5.5d is very low. It is a direct effect of the low object reflectance when compared to the red and yellow peppers in the image, as can be seen in Figure 5.5a. This is unintuitive to human observers, who have a high sensitivity for green (Figure 5.5c). We would expect the false-color display to better distinguish the green pepper from the background.

Unsupervised false-color display solutions that can tackle this problem are based on nonlinear dimensionality reduction techniques. However, the kernel trick, e. g. kernel-PCA, is typically not suitable for this application. It results in significantly more than three bands that need to be considered [Fauv 06, Fauv 09]. Most prominent in this field is ISOMAP [Bach 06]. It seeks a manifold coordinate system that preserves geodesic distances in feature space, as discussed in Section 3.1.2 on page 30. False-color images are created based on coordinates. Unfortunately, even after several improvements to algorithm complexity were made, including the use of approximation, ISOMAP still takes minutes to hours to compute [Bach 06, Cui 09]. Thus, fast, unsupervised, nonlinear dimensionality reduction for generation of a false-color display is an open problem.

5.2 False Coloring via Manifold Learning

In this section we describe how we can utilize the Self-organized Map (SOM) manifold-learning technique described in Section 3.1.3 for the purpose of a high-quality false-color display. As previously mentioned, a major application of the SOM is in fact data visualization. Several well-developed SOM-based techniques exist [Vesa 99]. These visualize the data in the layout of the neuronal network, e. g. the U-matrix that encodes a neuron's distance to its neighbors and helps to



(a) true color **(b)** [Gorr 12] **(c)** [Gorr 12] on spectral gradient

Figure 5.6: Visualization of *Fake and Real Peppers* using the method of Gorricha and Lobo.

manually identify clusters in the data [Taka 01, Tasd 09]. If the input data has a spatial context, a color-coding of a 1-D or 2-D SOM may be back-projected.

A mapping back to the spatial layout of a multispectral image was first proposed by Manduca [Mand 96]. They use a 1-D SOM to produce a grayscale display. This concept can be extended to a 3-D SOM for color-coded visualization of data in its original spatial layout [Vill 03]. More recently, Gorricha and Lobo train a SOM on geo-referenced data to color geographic elements, e. g. weather statistics or economical indices for certain areas on a map [Gorr 12]. Fonville et al. compare a 3-D SOM with other data mappings for visualization of mass spectrometry imaging data [Fonv 13]. In the spirit of said methods, we can create a false-color image in the hyperspectral domain by finding the BMU $\boldsymbol{m}_c$ for each image pixel $\boldsymbol{x}$ (Eq. 3.9). Then, an r, g, b triplet is created by scaling $\boldsymbol{r}^{(c)}$ by $1/n'_M$, where n'_M is the side length according to Eq. 3.10 (page 33). However, this method leads to suboptimal visual quality and achieves a significantly lower entropy than PCA false coloring, which hints at less information being conveyed. The reason is the limited amount of model vectors, resulting in a strongly quantized output. For example, in the configuration of Gorricho and Lobo, only 64 different color values could be produced [Gorr 12]. Figure 5.6 shows the resulting display when applying the method on a multispectral image.

We differ in two ways for a high-quality visualization of both multispectral and hyperspectral images. One, we train considerably larger SOMs. Two, we apply the ranked BMU lookup that was introduced in Section 3.2.2, as explained below. Note that increasing the SOM size alone is not a sufficient measure. While we use SOMs of size $n_M = 10^3$, we are still far from the capabilities of an 8-bit color display. For this, a map of significantly larger size would be needed, e. g. $n_M = 256^3$. Training such a SOM would become infeasible both due to longer learning times and the limited amount of available training samples.

5.2.1 Ranked BMU Lookup for False Coloring

As explained, a general drawback of the SOM dimensionality reduction is the strong quantization of the input space based on the relative small number of model vectors

n_M. In Section 3.2.2 on page 37, we propose a solution to the quantization problem. Recall the representative location $\boldsymbol{r}'$ given by Eq. 3.16 on page 37,

$$\boldsymbol{r}' = \sum_{j=1}^{n_C} w_j \cdot \boldsymbol{r}^{\left(c_j^{(x)}\right)} , \tag{5.5}$$

where $\boldsymbol{c}^{(x)}$ is a vector of BMU indices, sorted according to the distance of each BMU to the query vector $\boldsymbol{x}$. We calculate the weight vector $\boldsymbol{w}$ using the geometric progression in Eq. 3.19. We finally obtain

$$\mathtt{r} = \frac{r'_1}{n'_M} , \quad \mathtt{g} = \frac{r'_2}{n'_M} , \quad \mathtt{b} = \frac{r'_3}{n'_M} . \tag{5.6}$$

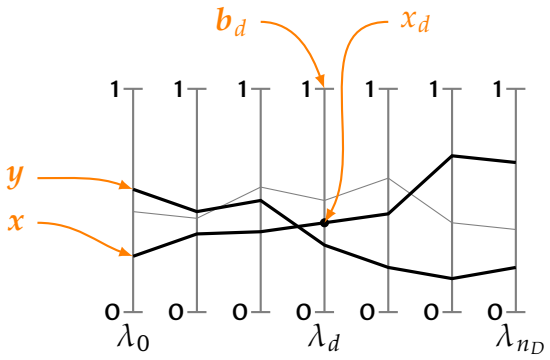
The resulting color display is supposed to, in general, convey the most relevant information, based on the learned manifold, without quantization artifacts. We expect it to be able to provide significantly more insight than a true-color or linear false-color visualization. However, due to being constrained to the spatial layout of the image, the underlying spectra remain hidden from the user.

5.3 Spectral Distribution Plots

As was discussed in the previous sections, most methods for visualizing multispectral or hyperspectral images, such as scatter plots or false-coloring, rely on dimensionality reduction. While the visualization of reduced data is helpful in many applications, it is hard to preserve subtle details. In contrast, a good visualization of the original data helps the observer evaluate how well a dimensionality reduction method fits a specific application. Existing approaches on depicting a multispectral image in its entirety are limited by the spatial layout of the image. The image data is modeled as a cube, with the z-axis corresponding to spectral bands, which are stacked on top of each other. Use of modern volume rendering techniques can make this representation useful in some scenarios [Li 08]. However in most cases, where there is no sparsity in pixels of interest, a very cluttered view is obtained that reveals the shortcomings of a simple 3-D arrangement.

One way of addressing this issue is to defer from the spatial relations in the image, for the time-being, and concentrate instead on a graspable representation of the spectral distribution. To do this efficiently, we employ the *Parallel Coordinates* method as explained below [Jord 10]. Parallel coordinates visualization was popularized by Inselberg and is a well-established technique for visualizing high-dimensional geometry and analyzing multivariate data [Inse 90, Hein 13]. It has been widely used, for example, in financial applications, life sciences, and geographic information systems. One can see the traditional spectral visualization as a specific instantiation of a static parallel coordinates visualization. By building on the more general concept, we can incorporate tools from the visualization community for high-dimensional data presentation. Moreover, parallel coordinates are part of an interactive visualization concept. Through interactive manipulation of the plot and synchronized other displays, we can overcome some limitations of the 2-D color display.

Parallel Coordinates



According to the parallel coordinates concept, a n_D-dimensional feature space (resulting from n_D spectral bands) is projected onto a two-dimensional view as follows. n_D parallel vertical lines denote the n_D axes, i.e. the n_D spectral bands. The y-coordinate on the dth axis corresponds to a spectrum's value at band d. To display the spectral vector of a pixel, a polyline is drawn with its vertices lying on the corresponding vertical axes.

The resulting display follows the layout of a plot where the x-axis would denote wavelength, and the y-axis denotes intensity. There are two important factors that enhance the usability of a parallel coordinates plot. One, the application of color and alpha to allow a concurrent view of large amounts of data points, and differentiate between them. Two, interactive manipulation of the plot, e. g. the dynamic highlighting of selected data points. Several useful extensions were proposed for parallel coordinate plots [Hein 13]. They include the rendering of curves instead of lines, rendering bundles of data with polygons, or density-based parallel coordinates. In the latter case, kernel density estimation can be employed or an implicit encoding of line density through the proximity of lines in the plot. As parallel coordinates best reveal the relationship between two adjacent axes, axis reordering is another prominent topic [Hein 13]. In our case, algorithms for band selection in hyperspectral data could be paired with the plot.

5.3.1 Optimized Parallel Coordinates

Drawing a polyline for each single multispectral vector has several drawbacks: it is time consuming; and the display may easily get cluttered in which case single polylines may not separate well from the rest of the data. A solution to both clutter and speed concerns is to draw fewer polylines, where a polyline can represent a set of pixels [Hein 13]. With this distinction, polylines that represent more pixels appear stronger.

We realize this representation by introducing a histogram in the feature space with n_B evenly distributed bins in each dimension. n_B is user-adjustable between 2 and the dynamic range of the captured data (typically 2^8 to 2^{14}). For example, the histogram for a 31-band multispectral image with $n_B = 256$ would hold ${n_B}^{n_D} = (2^8)^{31} = 2^{248}$ bins. Building a dense histogram of this size is not feasible, however a sparse histogram can be created by using an ordinary hashing algorithm. The key idea here is that the amount of occupied bins is bound by the spatial resolution of the input image. For example, a spatial resolution of 512×512 leaves only 2^{18} possible distinct values, effectively giving an upper bound to the amount of bins filled in a sparse histogram, or the amount of hash-keys needed.

For each populated bin, a polyline is drawn in the parallel coordinates visualization. The strength of the polyline is manipulated by assigning it an opacity α. It is

determined by the relationship between the number of pixels represented by the bin, n_K, and the total number of processed pixels, n_X,

$$\alpha = 0.01 + 0.99 c_\alpha \frac{f(n_K)}{f(n_X)}\,, \tag{5.7}$$

where c_α is a user-adjustable factor and $f(\cdot)$ is either a linear or a logarithmic function. The logarithm emphasizes bins with fewer pixels. The idea is that even a single pixel should be perceptible. The logarithm also ensures that the resulting dynamic range of alpha values can be represented reasonably well with an 8-bit alpha channel. With more recent graphics adapters, drawing onto a floating-point framebuffer is typically fast enough and the user can choose $f(\cdot)$ to be linear. A linear $f(\cdot)$ and lower value fo c_α are more adequate when many data points are shown.

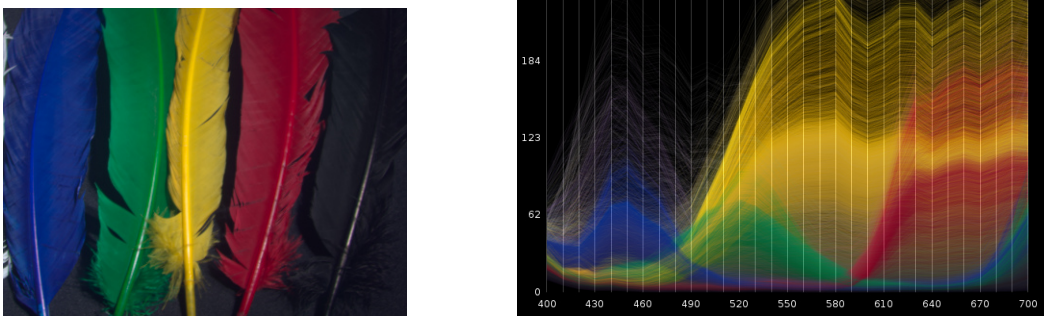
5.3.2 Drawing Refinements

We incorporate several measures to further enhance the visual quality and accuracy of the spectral distribution display [Jord 16b]. By dividing the feature space into a fixed number of equally spaced bins, the histogram applies a non-adaptive quantization of a spectral vector $\boldsymbol{x}$. A possible strategy to reduce the introduced quantization error is to employ a binning that is adapted to the observed data. A straight-forward method is to perform a separate histogram equalization in each dimension i, which enforces a uniform PDF of the mapped vector values x_i [Gonz 08, Chapter 3]. While it works well for big clusters of similar pixels, spectra that are sparsely represented in the image will suffer from such an operation. It may increase the average accuracy at a great expense in the precision of single pixel representations, which is not desirable.

We employ an alternative strategy by adjusting how a bin is drawn to achieve an improved general accuracy without a significant expense in the accuracy for any single pixel. When drawing a bin, we draw the mean vector of its contents instead of its mid-range values. This can be computed on-the-fly while adding new vectors, with a final normalization based on the number of entries. Figure 5.7 shows an example of the visual quality gain. We see that a broad description of the spectral distribution is possible with a low n_B.

Another hindrance in visual quality is the mutual obstruction of pixel representations. In many use-cases, pixels are color-coded (see Section 5.4.1). This involves effectively drawing several distributions on top of each other. In highly populated intensity ranges it can lead to extensive occlusions. We significantly reduce clutter by drawing the data in a globally shuffled order, i. e. the distributions of all labels appear intertwined.

More sophisticated methods were proposed for clutter reduction in parallel coordinate plots that would also apply to our case [Hein 13]. One approach is to only render a randomly sampled subset of the data points [Elli 06]. Others are based on drawing curves instead of lines, which removes disambiguities. Clutter can then be reduced by bundling the curves, e. g. based on data clustering [Hein 12, Palm 14].



(a) true-color display **(b)** refined drawing, $n_B = 256$

(c) simple binning, $n_B = 20$ **(d)** refined drawing, $n_B = 20$

Figure 5.7: Spectral distribution view of cropped *Feathers* image with different bin parameters. Spectra in true color.

Zhou et al. define energy terms on the control points of the curved lines for deriving visual clusters [Zhou 08].

5.4 Graphical User Interface

In our software framework coined Gerbil we combine the aforementioned visualization techniques and analysis tools. It addresses the need for an interactive visualization framework that is both sufficiently general for a broader range of applications and more versatile than existing basic representations. In our framework, initially introduced in 2010 [Jord 10], we follow a novel concept that enables an entirely new workflow in exploring a multispectral image [Jord 16b]. It revolves around presentation and exploration that makes the image data more apparent to the user, effectively allowing a more direct interpretation of the data. The user does not rely on, but is merely guided by, automatic processing.

Figure 5.8 depicts the main components of the graphical user interface: One, the visualization of original spectra (Figure 5.8 (1)) and the spectral-gradient spectra (Figure 5.8 (2)), both via parallel coordinates. Other available image descriptors for such displays are the L_2-normalized spectra and spectral vectors transformed by the PCA. Two, the spatial views (Figure 5.8 (3), (5)), including the depiction of single bands in each feature space, true coloring, false coloring, and an interactive similarity map. The interaction between these elements is augmented by tools that

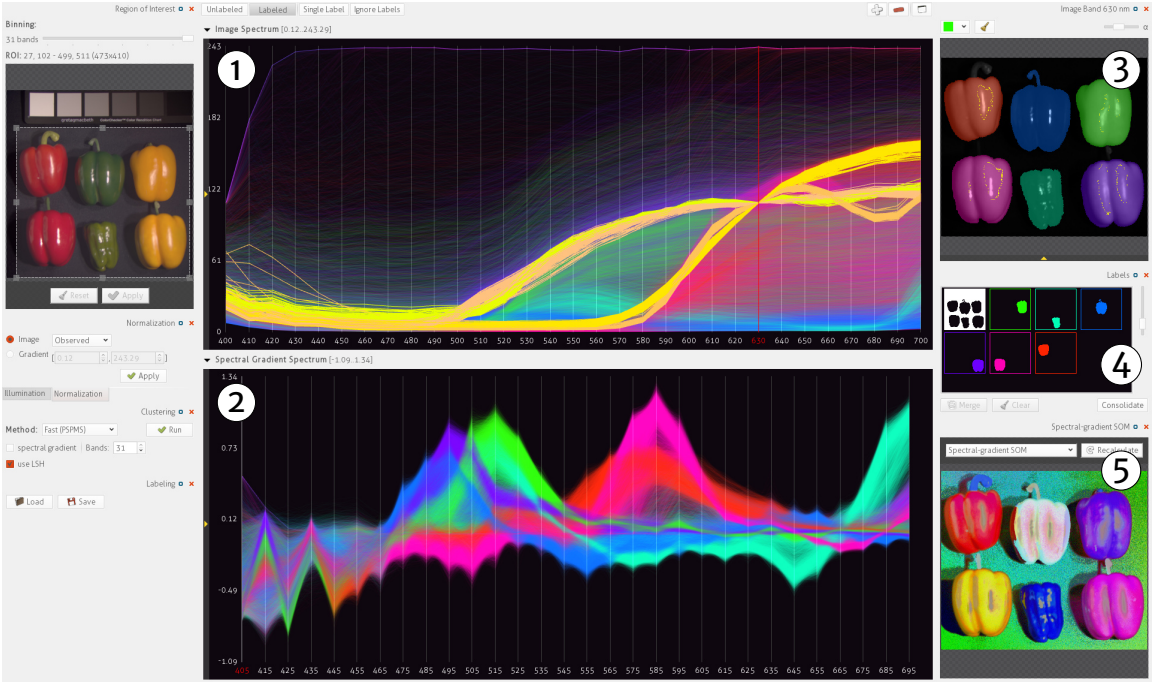


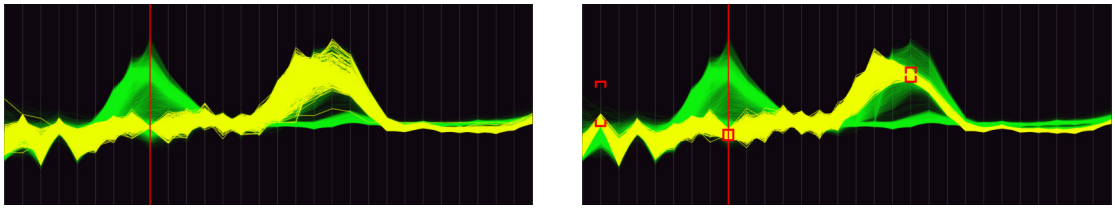
Figure 5.8: Gerbil user interface. (1) Spectral distribution view, (2) spectral gradient distribution view, (3) spatial view of a single image band with label overlays, (4) label manager, (5) false-color display.

give further guidance to the user, e. g. the feature space clustering and supervised segmentation techniques that were introduced in Chapter 4.

5.4.1 Interactivity

An important aspect of today's visualization approaches for multivariate data is interactive manipulation of the presentation. A single view most often cannot provide the full understanding that may be gained by a series of user-controlled depictions. User input is vital to parallel coordinates in particular. We provide several mechanisms for both transient (cursor highlights) and persistent (color labels) interactive viewing. In the parallel coordinate plots, the user can dynamically highlight specific data points, i.e. spectral vectors that represent pixels. These are displayed in yellow and with full opacity as an overlay over non-highlighted spectra. We realize this in OpenGL with layered frame buffers. Updates to the highlight only need a redraw of highlighted spectra. While the highlight constantly follows the keyboard and/or mouse input, the corresponding pixels are instantly highlighted in the spatial view. While scrolling through the spectra, the spatial view always reveals which pixels contribute to the highlighted spectra. This practice is known as *brushing and linking* in the literature [Hein 13].

Two modes of operation exist for highlighting in the spectral distribution: single-band limited and multi-band limited. The single-band limited highlight is formed by all spectral vectors falling into bins that share a specific intensity range in one band (see Figure 5.9a). The coarseness of this selection is therefore directly related to the binning parameter n_B. The user selects the band and intensity range via mouse-click



(a) single-band limited **(b)** multi-band limited

Figure 5.9: Highlighting modes used in the spectral gradient distribution view of a scene with two objects. In (a), one object is highlighted by a single intensity range selection in one band. In (b), further discrimination is achieved by adding intensity bounds in two further bands.

or cursor keys. The multi-band limited highlight provides a higher level of control in exchange of more detailed user input. Here, in each band separate lower and upper intensity bounds can be set (see Figure 5.9b).

Another method of highlighting exists in the spatial views. Using the mouse, the user can direct a cursor over individual pixels. The cursor is tracked to highlight the respective pixel under the cursor in the other displays. In each spectral distribution display (e. g. spectral gradient, PCA), the shape of the pixel in the respective descriptor is presented as a yellow overlay.

While dynamic highlights give instant feedback, they constantly change as the user investigates the data. It is often desired to keep a selection of pixels distinguished from the rest of the data, e.g. for comparison purposes. We call this pixel set a label; each pixel can be part of at most one label. A label can be seen as a permanent highlight. For each label, a distinct sparse histogram is created as described in Section 5.3.1. It is then drawn in the label color. When a histogram bin is part of the current transient highlight, the color is significantly shifted towards yellow. When the user has selected pixels within the transient highlight, they may add this set of pixels to a label or delete their label association. By doing so, they can iteratively refine the labeling of the data to concentrate on specific details. Another way to alter the labeling is to use a 'label' brush in the spatial view, or use automated segmentation methods as discussed in Section 4.2 and Section 4.3. Labelings can also be stored for later use. Label colors are automatically assigned for best visual discrimination. A preset of label colors consists of the primary and secondary palette, excluding yellow, which is reserved for temporary highlights. For more than five labels, colors are selected in the HSV color space. We divide the hue range equidistantly according to the number of labels, excluding yellow, while saturation and value components are set to 100 %.

Labels are important because they serve as a memory in the connection between different representations of the multispectral image such as between a single band and the spectral distribution. A selection, or temporary highlight in one representation is instantly propagated to the others. By labeling this highlight, it becomes permanent. The user can then continue their investigation within another data representation that may reveal new insights within this labeling. For example, a user may start by hand-labeling parts of a scene in the spatial view. Then, they may restrict the spectral distribution view to this label, or use the label to initialize a

multi-band limited highlight. In this operation, the limits on each band are set to include all pixels contained in the label. This helps both in finding similar spectra that are not yet included in the label and in separating several clusters within the label by adjusting the limits. The remaining selection can then be added to another label or form a new label.

As a result, we facilitate a workflow of inspecting an image that is not possible with existing hyperspectral analysis frameworks. It is based on concurrent, concerted work with both spatial and spectral displays, and allows a smooth and instantaneous switch in attention between them. Such a step-by-step exploration enables the user to quickly discover and grasp underlying information. In the visualization domain this procedure is considered a valuable tool for understanding complex data [Fuch 09].

5.5 Experimental Results

In this section, we will first evaluate the proposed false-color visualization based on manifold learning. We then examine our parallel coordinates visualization, with an emphasis on how a sacrifice in resolution for the sake of drawing time affects the output quality. We conclude with a discussion of image data descriptors based on our visualization capabilities and a small example for the proposed workflow in examining reflectance properties of a multispectral scene.

5.5.1 False Coloring

Several criteria have been proposed for judging the quality of a false-color display [Jaco 05]. An important measure is the relationship between the spectral difference of two pixels $\boldsymbol{x}$ and $\boldsymbol{y}$ and their color difference in the display. We discussed the SOM's distance preservation in Section 3.4.3 on page 51. Jacobson and Gupta consider the spectral angle between spectra and the color distance in a perceptually uniform colorspace (CIE Lab). In general it is debatable which distance is most relevant in both spaces. They further describe the goals of a natural palette and color symbolism, effectively that a false-color display should produce colors similar to true color, and also in other literature, agreement of a false coloring with the true coloring is measured [Zhu 07]. This is conflicting when metamers are present in the scene. It also becomes less practical when the wavelength range of the hyperspectral image diverges from the range of visible light. Other goals are edge preservation and the discriminability of different spectra in the visualization versus smallest effective differences. The latter describes that visual distinctions should be no larger than needed to effectively show relative differences. The SOM-based false coloring is expected to be reliable in the goal of edge preservation based on our edge detection results. However, it is not expected to perform best under the criterion of smallest effective differences, which can also be contradicting to spectral discriminability.

In this evaluation, we will concentrate on the discriminability of different spectra. The simplest statistic in this context is the variance of the output, which is directly related to the image contrast. Images with higher variances have a better contrast,

which makes visualization simple and appealing [Kotw 13]. Naturally, a high discriminability also goes with a wide range of color shades used in the visualization. From information theory, we know entropy as a measure of the information content in an image, in our case the false-coloring display [Zhu 07, Kotw 13]. Entropy was defined by Shannon as the lower bound on the number of bits needed to communicate the state of a random variable [Bish 06, Chapter 1],

$$\mathrm{H}(x) = -\sum_{x} p(x) \log_2 p(x) \,, \tag{5.8}$$

where $p(x)$ is the probability of observing a value x. For the case of $p(x) = 0$ we define $p(x) \log_2 p(x) = 0$. We regard each of the $\mathtt{r}$, $\mathtt{g}$, $\mathtt{b}$ components in the false-color display as a sample distribution. Furthermore, we work with 8 bit images and normalize the entropy values accordingly, to obtain

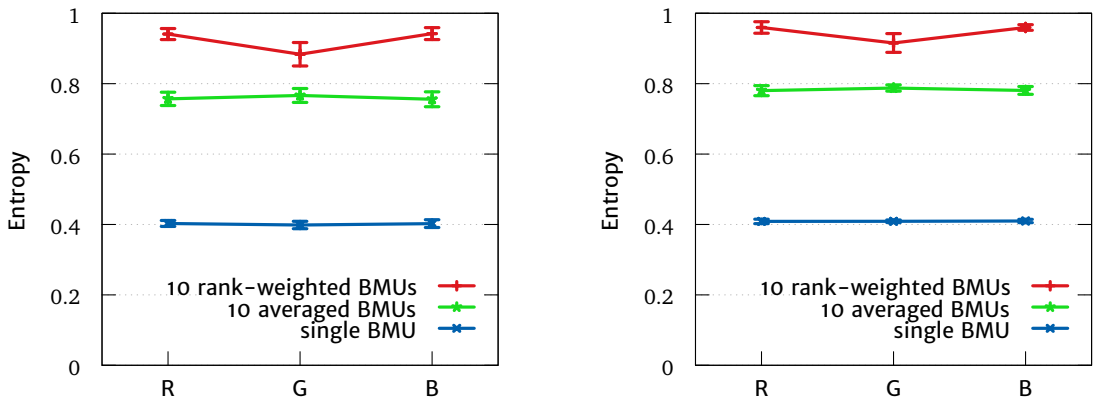
$$h_{\mathtt{r}} = \frac{\mathrm{H}(x_{\mathtt{r}})}{8} \,, \quad h_{\mathtt{g}} = \frac{\mathrm{H}(x_{\mathtt{g}})}{8} \,, \quad h_{\mathtt{b}} = \frac{\mathrm{H}(x_{\mathtt{b}})}{8} \,. \tag{5.9}$$

Intuitively, a high entropy value for an image component (i. e. an intensity map) means that the intensity value observed at any position in the channel is hard to predict. This corresponds to the richness of information, resulting in a more meaningful representation [Zhu 07]. Entropy is a better measure than variance or contrast, which can also be misleading in qualitative evaluation. A false-coloring method that produces a high contrast image does not necessarily convey more information than a low-contrast counterpart, e. g. if the image consists mostly of extreme values. A false coloring is good if, in accordance with the original data distribution, many different color accents are produced. Note however that the highest entropy is achieved with completely random intensities. Entropy by-itself is not a measure of how useful the observed information is. We regard it in combination with the results in Section 3.4.3 and a qualitative interpretation of the output.

Ranked BMU Lookup

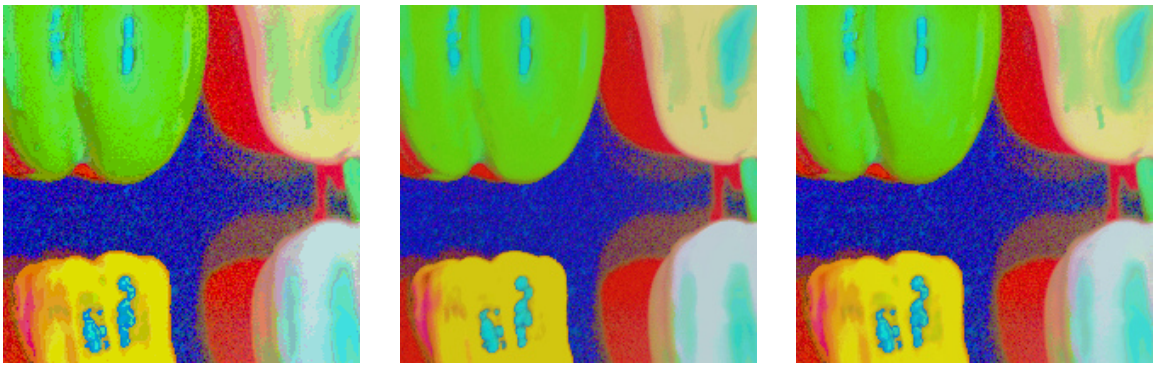
First we examine the influence of the ranked BMU lookup on visualization quality. For this, we train a SOM with $n_M = 1000$ model vectors and apply three kinds of lookup for coloring a pixel: One, a simple single-BMU lookup. Two, a lookup of 10 BMUs with flat weighting (effecting an unweighted average). Three, 10 BMUs with the proposed rank weighting. We compute the entropy from the resulting false-color images for all images from the *CAVE* and *Foster* dataset. Recall that the former represents a controlled lab environment while the latter is comprised of natural scenes. Remote sensing images will be considered later in the experiment when we compare our method with PCA.

Figure 5.10 depicts the average entropy values with standard deviation for both datasets. We observe that in the case of the single-BMU lookup, the quantization of the SOM results in weak entropy levels, suggesting that less information is conveyed than theoretically and practically possible for this visualization format. The use of several BMUs provides more diversity in the displays. The ranked-BMU lookup appears strongest in this test, achieving very high normalized entropy values, considering the theoretical maximum of one (corresponding to eight bits).



(a) *CAVE* dataset **(b)** *Foster* dataset

Figure 5.10: Entropy values obtained in the three components of SOM-based color visualizations.

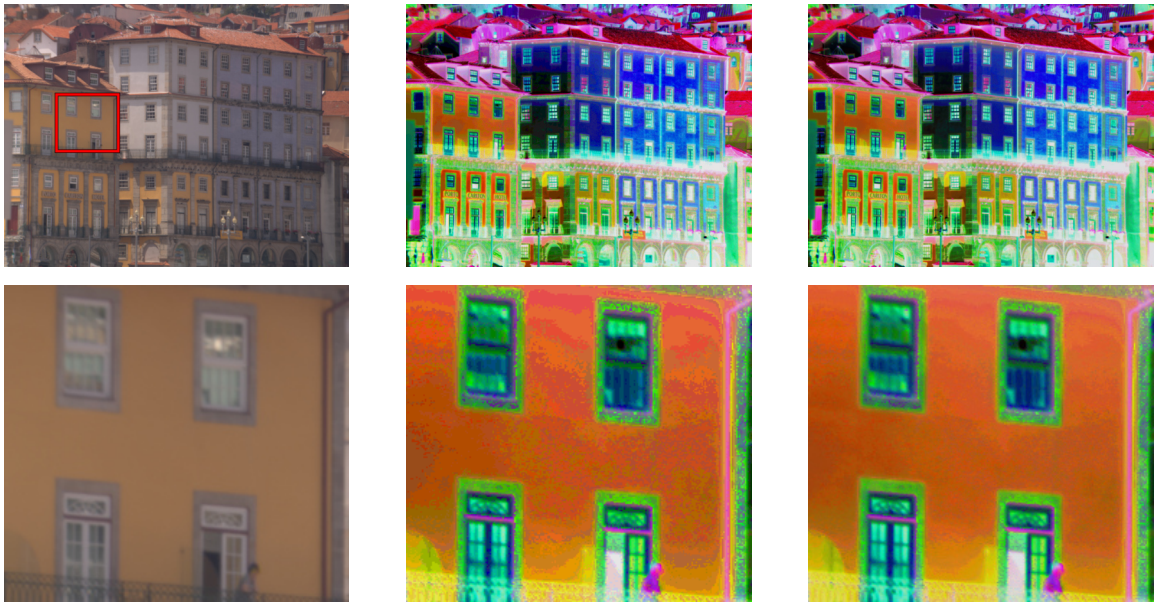


(a) single BMU **(b)** 10 averaged BMUs **(c)** 10 rank-weighted BMUs

Figure 5.11: Comparison of BMU lookup on *Fake and Real Peppers* image. Results are shown for a detail of the image for better visibility. The full result of (c) is depicted in Figure 5.14c.

These results match the impression given by a visual comparison of the false-color outputs. Figure 5.11 shows results on a detail of the *Fake and Real Peppers* image, computed from the spectral gradient of the image. As expected, the traditional lookup of a single BMU depicted in Figure 5.11a captures different materials, specular highlights and shadows well, but suffers from quantization effects. When adding more BMUs as in Figure 5.11b, a smooth display is obtained at the expense of significant details. For example, the shadowed areas of the pepper depicted in green are almost lost. Finally, the proposed rank-weighting scheme displayed in Figure 5.11c preserves most detail of all three methods in an artifact-free presentation.

Figure 5.12 shows the SOM false-coloring for an image from the *Foster* dataset. In the detail, we see the strong quantization artifacts of a traditional SOM lookup being eliminated while retaining sharp details present in the original data, which otherwise could be hard to distinguish from artifacts.



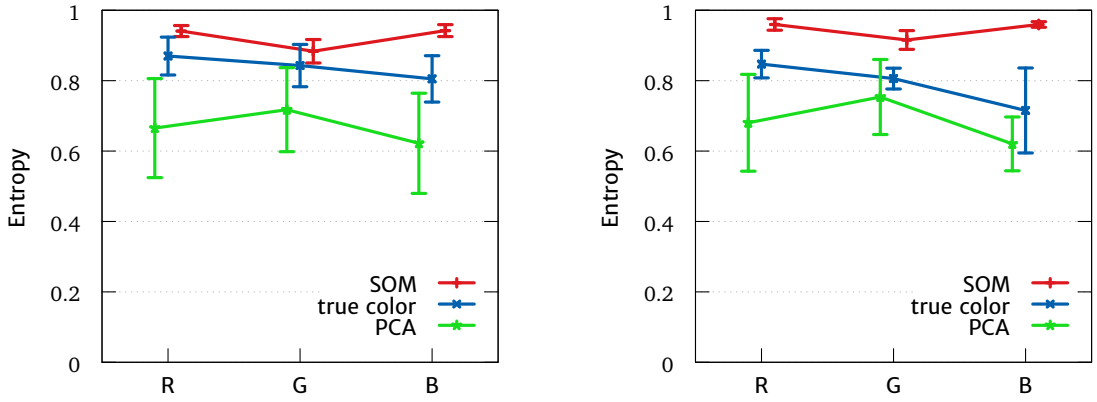
(a) true color **(b)** SOM with $n_C = 1$ **(c)** SOM with $n_C = 10$

Figure 5.12: Comparison of BMU lookup on *Ribeira* image from the *Foster* dataset. (a) true color, the image detail below is marked in the full image in red. (b) and (c) visualizations obtained by using a single BMU lookup. and a ranked BMU lookup, respectively.

Comparison to PCA

Figure 5.13 depicts the average entropy values with standard deviation obtained by true-color mapping, PCA false color and SOM false color. Regarding the PCA, we see that the information contained in the three components appears unbalanced. This is expected, as the contrast sharply declines in the secondary PCA components (which form the red and blue channels). The entropy of a true color mapping is typically higher than the one achieved by PCA. Note that the true color mapping is not practical for depicting remote sensing data where relevant information is captured outside the visible wavelength range.

We now perform a visual comparison on three example images. Our goal for the image shown in Figure 5.14 is a visualization that overcomes metamerism and therefore needs to significantly contrast with true color. As was seen before, the task of material separation works best with the spectral gradient descriptor, which we also employ here for PCA and compare it to the performance on the original data space for SOM. We find that different materials of the peppers, three of which are plastic, are well separated by the SOM on the spectral gradient, but also, to a lesser extend in the original image space. You can discern plastic peppers from organic ones by looking at the coloring of their stems. Also, reflectance effects such as inter-reflections, specular highlights and shadowed regions are captured. The PCA visualization fails to distinguish the yellow peppers and the separation of the shadow on the backdrop is weak, even while operating on the spectral gradient descriptor. Note that PCA performance is even worse when operating on the original



(a) *CAVE* dataset **(b)** *Foster* dataset

Figure 5.13: Entropy values obtained in the three components of different color visualizations.

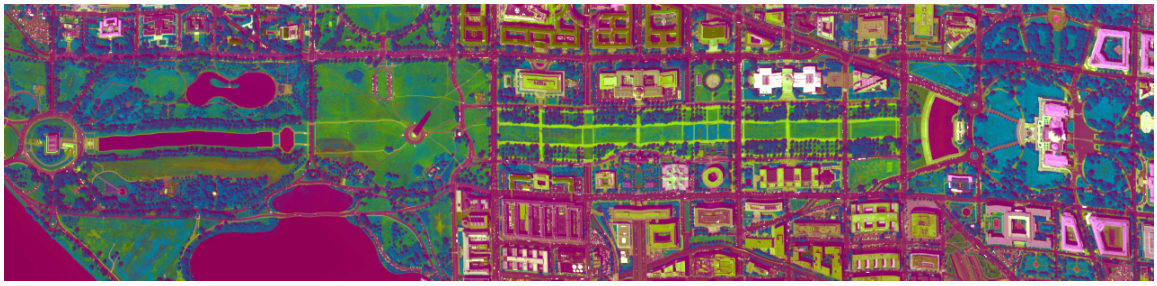
(a) SOM **(b)** PCA on spectral gradient **(c)** SOM on spectral gradient

Figure 5.14: False-color visualizations of *Fake and Real Peppers* image.

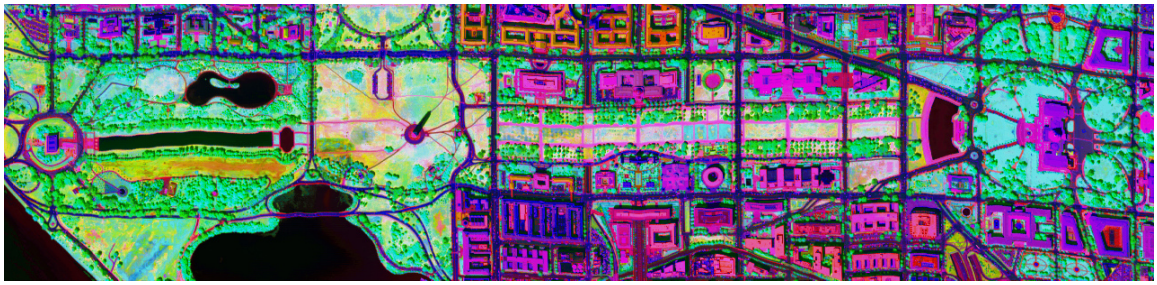
spectra, as was seen in Figure 5.5 on page 127. See Appendix D on page 178 for another example image from the *CAVE* dataset.

Figure 5.15a and Figure 5.15b depict the *D.C. Mall* remote sensing image colored by both the PCA and the SOM method. The SOM provides a good separation of the classes grass, tree, roof, road, trail and water. We can also read more subtle details from the visualization, such as different roof types or different grass segments. For this image, PCA also performs well. However, some structure is not as easy to distinguish, and some classes lack separation (e.g. water vs. road). In general we found that PCA often contrasts well a single class of pixels in the image (here a rooftop in white) at the expense of a good general contrast between other classes.

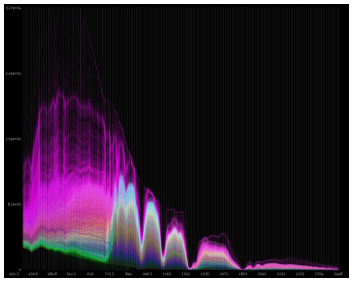
Note that the SOM itself can be visualized using this method during, as well as after, its training. For this, we combine the false-coloring with spectral distribution plots which then depict the distribution of the model vectors, and make it visually comparable to the distribution of the training data. These visualizations help in both assessing the training and in fine-tuning its parameters. The figure also depicts the SOM-colored spectral distribution view (Figure 5.15c) as compared to the spectral distribution of a SOM trained on the image (Figure 5.15d), as well as the ten



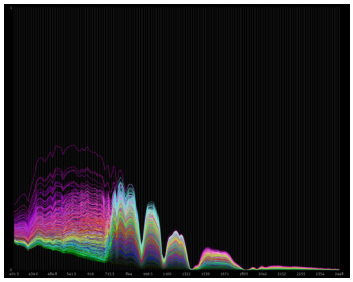
(a) image in PCA false-color, $(h_r, h_g, h_b) = (0.78, 0.76, 0.66)$



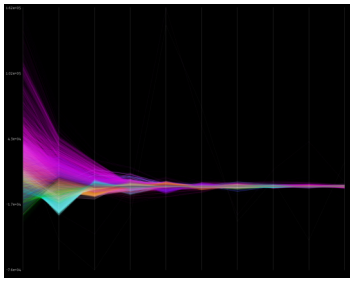
(b) image in SOM false-color, $(h_r, h_g, h_b) = (0.93, 0.92, 0.96)$



(c) original distribution



(d) SOM distribution



(e) image PCA distribution

Figure 5.15: False-color visualization of the *D.C. Mall* image and parallel coordinates visualizations computed from the data. Pixels in (b) and spectral vectors in (c), (d), (e) are colored using the SOM visualized in (d) according to Eq. 5.6.

principal components computed by PCA (Figure 5.15e). In this case, we can see the clear associations between colors and spectra, e. g. magenta for shiny roofs, green for trees, and a range between yellow and cyan for pasture and soil.

Figure 5.16 depicts results for the *Indian Pines* remote sensing image. In this example, PCA catches the steel tower in the upper right, but not more subtle differences between crop classes, effectively providing no benefit over the three-band composite in Figure 5.16a. While the SOM visualization illustrates the same effects as the other false-colorings, it also helps distinguish several of the classes shown in Figure 2.5 (page 13). This is illustrated by the average colors assigned to pixels from each class, as depicted below each visualization.

Reproducability

A peculiarity of the SOM false coloring is that based on the stochastic nature of the training process, the visualization can signficantly change in subsequent calculations,

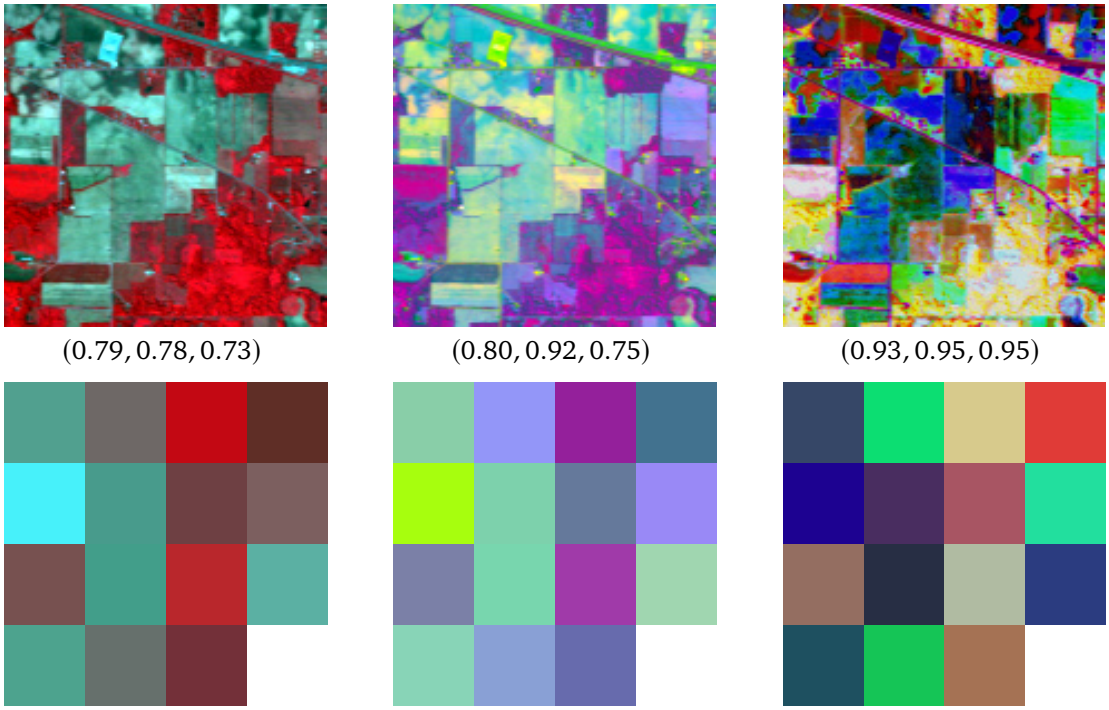


Figure 5.16: Color visualizations of *Indian Pines* image and average colors assigned to pixels from fifteen classes. $(h_\mathrm{r}, h_\mathrm{g}, h_\mathrm{b})$ is denoted below each visualization.

as was discussed in Section 3.2.3 on page 38. The main cause of this phenomenon is the rotational invariance of the SOM. This has no significant effect on the objective quality of the visualization, however it can be considered a drawback that colors are not predictable from one visualization to another, as is the case with true-color or PCA visualizations. In Appendix D on page 178 we show an example of a false-color visualization provided by a user of the software and how it relates to a previously published visualization result. In Figure 5.18b, some color mappings also significantly differ from Figure 5.14c, as the SOM was trained only on the depicted detail of the image. In consequence, it learned a different manifold which does not include spectra from other objects in the full scene.

We conclude this section with a note on the computational performance. The false-coloring scheme proposed here is based on training a cubic SOM with $n_M = 1000$. We found that with the ranked BMU lookup, good results are also obtained with considerably smaller maps, starting with sizes from $n_M = 216$. In this case, entropy values are reduced by about 10 %. After training, a single readout is needed for each pixel. The wall-clock times reported in Table 3.1 on page 50 therefore roughly apply here as well. For example, with an Intel Core i5-6600 CPU with four cores, false coloring of an image from the CAVE dataset can be performed with a small SOM in under a second, with a larger SOM in under three seconds. For the *D.C. Mall* image of higher spatial and spectral resolution, it takes under 3.5 seconds with the small map, and under 11 seconds with the larger map.

5.5.2 Efficient Parallel Coordinates

We showed spectral distribution plots throughout this thesis in qualitative discussions of other algorithms. In this study, we shed some light on how feasible an accurate plotting is in an interactive setting. We employ a histogram-based quantization of spectra for faster drawing. The error introduced by this approximation can be measured by the average root mean squared error (RMSE) as well as the maximum absolute deviation (MAD), between original vectors $\boldsymbol{x}_i$ and their quantized counterparts $\hat{\boldsymbol{x}}_i$. This gives us the measure

$$\text{RMSE} = \sum_{i=1}^{n_X} \frac{\sqrt{\frac{\sum_{d=1}^{n_D} (\hat{x}_{id} - x_{id})^2}{n_D}}}{n_X}, \tag{5.10}$$

for the average RMSE, and

$$\text{MAD} = \max(\max(|\hat{x}_{id} - x_{id}|, d \leq n_D), i \leq n_X), \tag{5.11}$$

where $\max(\cdot)$ is the sample maximum.

We evaluate RMSE and MAD for both the naive bin center and the refined vector mean drawing methods with varying n_B on several datasets from different application domains. The number of bins per dimension n_B is a crucial parameter. It lets the user choose between drawing speed, viewing quality, and accuracy. Even a considerably low n_B should provide acceptable accuracy, and the speed-up by lowering n_B be effective. We use a desktop machine equipped with a quad-core Intel Core i7 CPU running at 2.80 GHz, and a GeForce GTX 550 Ti consumer graphics card for testing the computational performance. We draw in WUXGA resolution and measure the time needed for drawing operations via `GL_TIMESTAMP` [Shre 13, Appendix H].

In Figure 5.17 we plot execution time against accuracy for varying n_B. RMSE and MAD are plotted on a logarithmic scale. We can observe that the average RMSE becomes negligible with higher n_B for both drawing methods. However, the refined method achieves low RMSE values for considerably lower settings of n_B. Due to outliers present in some of the histogram bins, the MAD for the refined method is somewhat higher than the original MAD.

The time needed to build the histogram is denoted as *Binning* and is not determined by n_B. The time needed for preparing the geometry and loading it on the GPU (*Loading*) slowly grows with n_B. In contrast, n_B plays an important role for the time needed for the drawing operation (*Drawing*). For higher n_B values, the time needed for drawing grows to multiples of the time needed for preparation. Hence the histogramming plays an important role in achieving interactivity.

In Table 5.1 execution time and accuracy measures for the refined drawing method are listed for the *CAVE*, *Foster*, and *D.C. Mall* datasets. Please refer to Table 2.2 on page 12 for dataset statistics. The times shown for preparation are the combined histogram building and geometry loading times. As in Figure 5.17, it is observed that a low n_B can already achieve a small quantization error. Setting $n_B = 64$ is a reasonable compromise between speed and accuracy on the tested

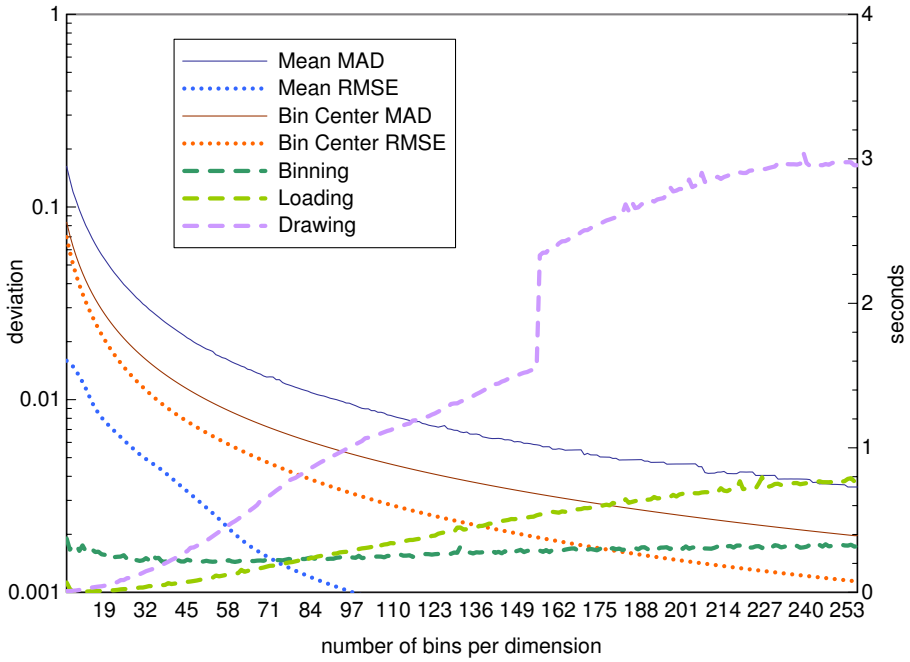


Figure 5.17: Spectral distribution plotting performance on a scene from the *Foster* dataset. Quantization errors are plotted for the naive (Bin Center) and the refined (Mean) drawing method.

Dataset	n_B	Preparation (s)	Drawing (s)	RMSE	MAD
CAVE	32^{31}	0.089	0.161	0.002	0.027
	64^{31}	0.115	0.240	0.001	0.014
	256^{31}	0.177	0.390	0.000	0.003
Foster	32^{33}	0.274	0.146	0.005	0.031
	64^{33}	0.358	0.554	0.002	0.014
	256^{33}	1.085	2.955	0.000	0.004
D.C. Mall	32^{191}	0.585	1.789	0.001	0.026
	64^{191}	0.881	3.171	0.000	0.013
	256^{191}	1.020	3.554	0.000	0.003

Table 5.1: Average drawing times and accuracy.

datasets. It provides an effective speed-up in comparison to a high histogram resolution without a perceivable loss in drawing accuracy.

It can be seen in Table 5.1 that on our test machine, even with a moderate setting, drawing the spectra from a large image may still take several seconds. Typically it is impractical to work on a high-resolution image without a region-of-interest. Yet, we alleviate longer drawing times by incrementally drawing the data (disabled in the benchmarking) in order to provide direct visual feedback in the form of a rough approximation of the full data (as pixels are drawn in a random order).

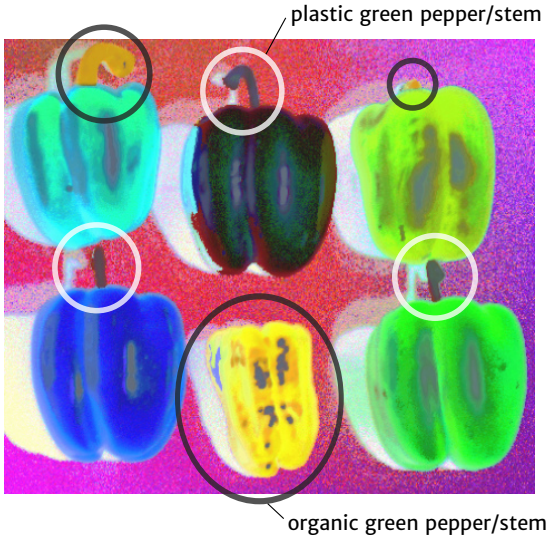
5.5.3 Combined Visualization and Image Data Descriptors

The two proposed visualization techniques, namely false coloring and distribution displays, go hand-in-hand. As was shown in Figure 5.15, the pixel coloring obtained through the SOM can not only be applied in the spatial domain, but also when visualizing the distribution of the image, or another image descriptor. While the spectral and spatial display paradigms work best together in an interactive workflow, we may illustrate their synergy here based on the shared color-coding.

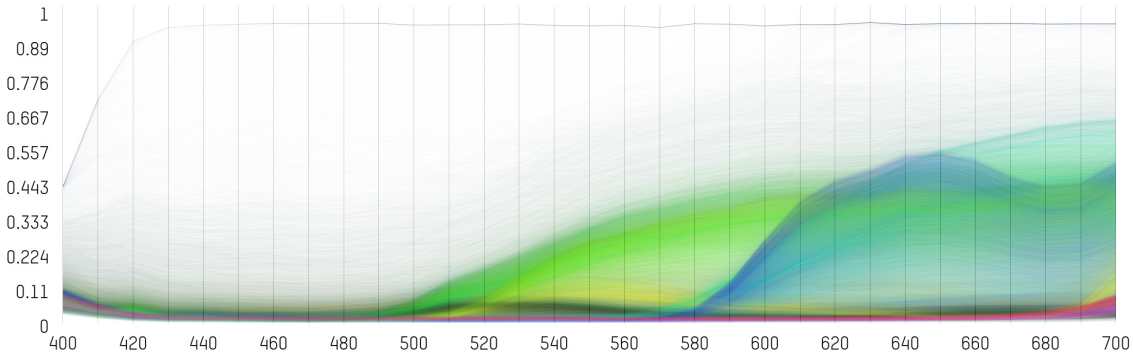
In the SOM false-color display computed on the spectral gradient, Figure 5.18b, we see that pixels with the same material or reflectance properties are consistently colored. One such example are the specular highlights, colored in light gray. Another instance is the stems of the plastic and real peppers as annotated in Figure 5.18b. Note that the peppers depicted in light blue, yellow/orange and light green are real, while the others are plastic. Figure 5.18c allows us to see the relationships between spectra of the six peppers in the image. The first prominent effect, as discussed before, is that the green peppers in the middle both elicit a very weak spectral response when compared to the yellow and red peppers. Furthermore, a 'dent' in the spectral response at around 670 nm to 690 nm distinguishes the fake red pepper from the real one. The distinction is less clear, but still visible, between the two yellow peppers to the right in Figure 5.18a, colored in two shades of green in the other figures. We see that the fake pepper reflects the light stronger in the range of 500 nm to 530 nm. The spectral gradient distributione depicted in Figure 5.18d makes the separation of these different materials easier for us. We see the two different peaks between 490 nm and 500 nm, and between 510 nm and 520 nm, for the fake yellow pepper, and the real yellow pepper, respectively. The real pepper stems and a part of the real green pepper are colored in orange, while the majority of the real green pepper is colored in yellow. The reason for this is revealed by the respective distinction in the spectra of these pixels at 630 nm to 650 nm and further on at 680 nm to 700 nm (both annotated with circles in Figure 5.18d). This distinction is easy to miss in the display of the original spectral distribution, Figure 5.18c.

This example illustrates that the choice of image descriptor is important for visual analysis of multispectral data. The parallel coordinate plots are a powerful way to access the spectral characteristics of the data. A false coloring can assist in deciphering the structure of the distribution revealed in such a plot. See Appendix D on page 179 for another example.

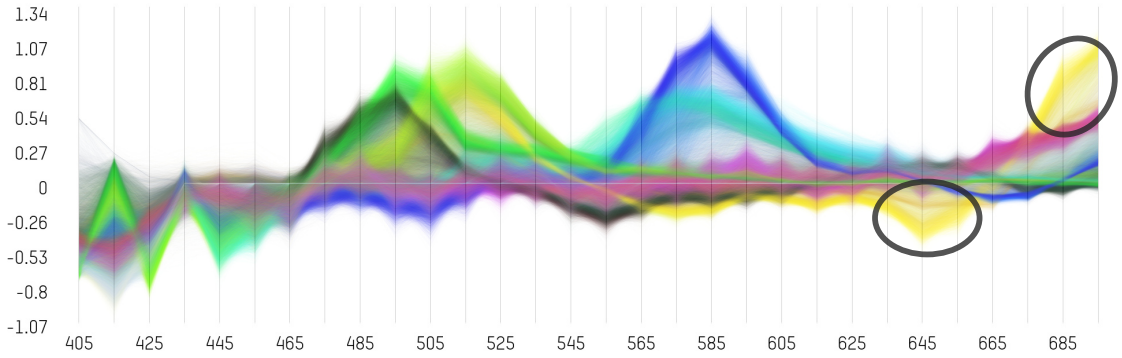
(a) true-color display



(b) false-color display



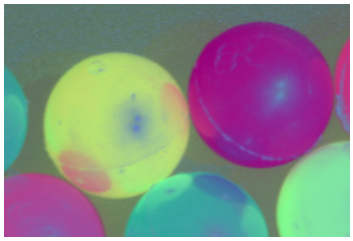
(c) spectral distribution



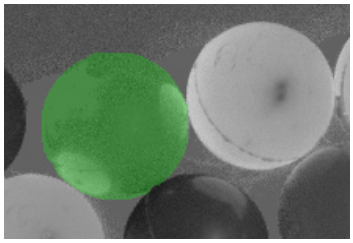
(d) spectral gradient distribution

Figure 5.18: Proposed Visualizations of *Fake and Real Peppers* image. In (b) (part (5) of Figure 5.8, color-inverted), the occurrences of two materials in their respective color coding (orange and dark gray) are annotated with circles. In (d), two distinctions between differently colored spectra originating from the real green pepper are annotated with circles.

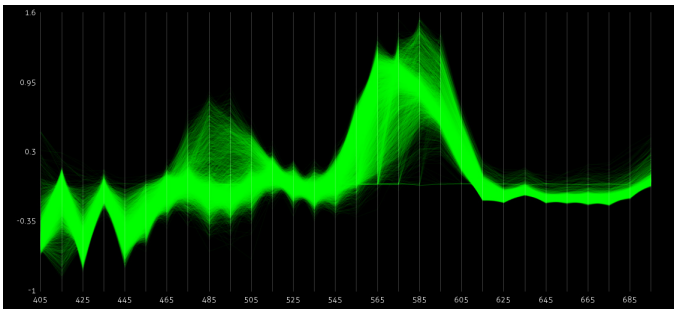
(a) true color



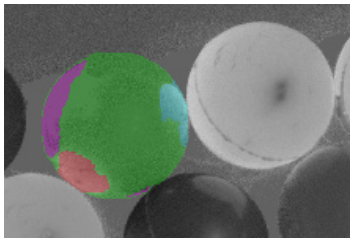
(b) spectral gradient PCA



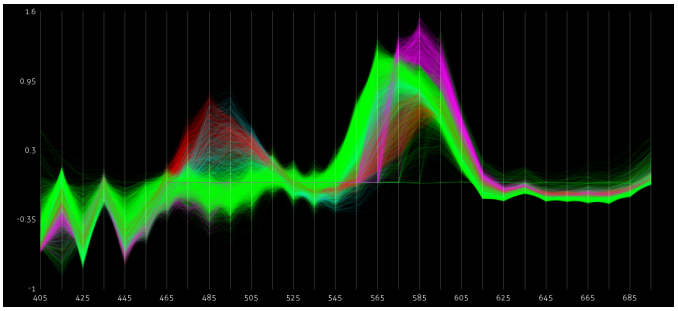
(c) first labeling



(d) spectral gradient distribution of (c)



(e) second labeling



(f) spectral gradient distribution of (e)

Figure 5.19: Inspection of cropped *Superballs* image. Labels are shown as overlay on the spectral gradient at 490 nm to 500 nm.

5.5.4 Example Workflow

For an example on how the different displays work together with automated analysis tools, we revisit the case of the *Superballs* image that was previously discussed in Section 4.2.4 (page 108). It shows plastic balls of various color that highly reflect on each other. In Figure 5.19, the user investigates a red ball specifically. With an RGB camera, we could only spot the reflection from the yellow ball (see Figure 5.19a). Calculating the PCA on the spectral gradient also reveals the reflection from the green ball (see Figure 5.19b). However, the blue balls also reflect on the red ball, which is not revealed by both depictions. The first labeling, Figure 5.19c, is obtained by supervised segmentation: One foreground seed is set in the middle of the ball, and a circle is drawn around the ball for background seeds. When looking at the spectral gradient distribution of that segment (see Figure 5.19d), we see the deviations from the distribution caused by these inter-reflections. Browsing through these parts of the spectral gradient distribution plot with the interactive highlight reveals their spatial locations. By setting corresponding seed points for new supervised segmentations,

we can find three additional segments within the ball (see Figure 5.19e). The spectral gradient distribution plot (see Figure 5.19f) reveals how the pixels from these three segments contribute to the deviations in the distribution. To further understand them, the user might continue by also segmenting the surrounding balls, effectively adding them to the plot.

5.6 Discussion

In this chapter we discussed various approaches on visualization of multispectral and hyperspectral images, and how these can be combined for interactive analysis.

The evaluation of the SOM false coloring supports our expectation that it performs well for visual discrimination of spectra. As the SOM topology is fitted on the observed data manifold during training, said data then fits well into the low-dimensional data space formed by the SOM topology, in this case a cube that provides `r`, `g`, `b` coordinates. This leads to a high contrast, but also high entropy in the resulting display, given that we employ the proposed ranked BMU lookup technique. Without, the quantization of the SOM would significantly lower the amount of information conveyed in the image. The false-color display is especially helpful for selecting regions-of-interest (ROIs) that should be examined. In large images, due to clutter and computational expense, it is beneficial to select a ROI before performing further visualization tasks. Additionally, the false-color display can give a good first impression of the data, e.g. to spot the same reflectance in several regions of the image or to find inhomogeneities within a depicted object.

We introduced spectral distribution plots based on parallel coordinates to make the actual distribution of reflectances in the scene accessible to the user. Due to a fast and accurate drawing strategy, these plots can be embedded in a software framework that facilitates interactive exploration within the structure of the data distribution. For this, false-coloring and spectral distribution plots go hand-in-hand, as the coloring of spectra helps in quickly understanding the plot and identifying clusters visually. Furthermore, the proposed workflow is based on connecting the spectral and spatial views on the data. In our experiments, we showed that the combination of these techniques, and analysis tools discussed in Chapter 4, can be a powerful tool for general hyperspectral analysis.

The Graphical User Interface exemplified in this chapter provides new, intuitive ways for understanding a multispectral or hyperspectral image also for novice users. We anticipate that our software framework is of assistance especially in emerging applications of this imaging domain, as is indicated by our communications with current users.

Chapter 6

Outlook

In this work, we presented several aspects of image processing and visualization for interactive analysis capabilities in the multispectral and hyperspectral domains. As the scope of this work is necessarily limited, we have suggestions for future work in these areas to validate and improve our work, or use it as a basis for new algorithms.

Edge Detection We introduced two variants of a new pseudometric for edge detection in Section 4.1.2, referred to as SOM_1 and SOM_2. The latter is based on the observation that the direct distance between two spectra in a SOM is not always a sufficient edge criterion. SOM_2 confines the effect of varying cluster sizes by considering a larger set of best-matching units (BMUs). We might also tackle the variation in BMU locations more directly with an edge measure that compares intra-class and inter-class variation. Recall the merge criterion of Felzenszwalb and Huttenlocher for two adjacent superpixels. In the same vein, a measure SOM_3 could compute the ratio of intra-class variations (i. e. the variations within the two BMU sets) and inter-class variation (i. e. the variation in a joint BMU set). Early tests of such a measure look promising and seem to complement SOM_2.

While we did a qualitative comparison of SOM_1, SOM_2 and other measures for edge detection, quantitative analysis is lacking. It would be worthwhile to benchmark the methods on synthetic images using quantitative measures. Note that designing a meaningful dataset as well as choosing appropriate measures for a fair evaluation is not a trivial task [Lope 13].

Clustering In Section 4.2.2 and Section 4.2.3, we proposed various speed-ups of the FAMS mode-seeking algorithm. While mean shift in general can be performed with any radial-basis kernel, FAMS uses the Epanechnikov kernel and the L_1 distance for computing adaptive bandwidths. A reason for this is that the locality-sensitive hashing (LSH) employed in FAMS approximates nearest-neighbor search in L_1. As the `mssom` algorithm avoids LSH, it should be considered to perform nearest-neighbor searches in L_2 and also to try a different kernel, e. g. the Gaussian kernel.

More generally, as was discussed in our experimental evaluation, we need better benchmarking capabilities for clustering methods, based on valid ground-truth labels. Furthermore, the proposed methods need to be tested for more specific applications to assert their general applicability.

False-color Visualization Our approach to false coloring, detailed in Section 5.2, maps a cubic SOM topology onto the RGB cube. Note however that the RGB color space is not perceptually uniform, which is a drawback when communicating the distance on the data manifold through color coding. It should be beneficial to employ a mapping to a different color space that is designed for perceptual uniformity. An example would be the CIE 1976 (L*, u*, v*) color space. Note that this also changes the shape of the SOM topology.

The stochastic nature of the SOM results in a major drawback for false coloring. Spectra are assigned seemingly random colors, mainly due to the rotational invariance of the map. This effect might be lessened by utilizing the batch variant of the SOM algorithm [Cott 16]. This variant does not update the map in each step, instead, an accumulated update is performed after determining BMU and influences for all input samples. Note, however, that while this behavior results in a more predictable map, it does not necessarily find an optimal solution. Another possibility that is applicable to specific application scenarios is to integrate the proposed semi-supervised learning variant. In this case, the general map orientation can be fixed if several most-relevant spectra are known.

Spectral Distribution Visualization We discussed several extensions to the parallel coordinate display concept which can be of use for our application in Section 5.3. Especially interesting in this regard is the clutter reduction based on rendering spectra as bundled curves instead of straight polylines. The bundling of curves needs to be determined either in the feature space or in the plot. We proposed methods to very quickly obtain a data clustering, most particularly the `somtopo` method. This can serve as an input to the bundling algorithm, such that the clustering results can not only be used for a color coding but also for an uncluttered view of the data.

Spectral Unmixing The task of spectral unmixing is typically a two-step process, whereas in the first step, the constituent prototypes of a scene, referred to as endmembers, are found. In the second step, unmixing methods fit each pixel to a linear mixture of the endmembers. Spectral unmixing is an important tool in remote sensing [Biou 13], where the area of land represented by a pixel can span several square meters. However, we can find mixtures in pixels also in other application domains, e. g. medical diagnosis or cultural heritage. When analyzing historical paintings, the paint pigment mixtures used by the artist are of high interest. While spectral unmixing is out of the scope of this thesis, it can play a significant part in interactive analysis of hyperspectral data.

Current spectral unmixing algorithms face challenges in both endmember detection and unmixing steps. Recall the proposed fuzzy mean shift variant of `mssom` from Section 4.2.3, which is based on a rank-weighted BMU scheme. Note that fuzzy clustering in the hyperspectral domain is closely related to the problem of spectral unmixing: Each cluster is represented by a prototype (its mode) and each spectrum $\boldsymbol{x}$ has a set of weighted cluster assignments that sum up to one, which effectively can be translated to $\boldsymbol{x}$ being a mixture of cluster prototypes. It is a valid assumption that endmembers are modes in the distribution when a reasonably sized set of pure pixels is part of the scene, and these modes are found by `mssom`. Commonly

made assumptions about endmembers could be employed to further reduce the set of modes. In traditional unmixing algorithms, when finding a linear mixture of prototypes, it is hard to fulfill the simple, but crucial constraints that all mixture weights are non-negative and sum up to one. However, in our rank-weighted cluster assignments, these properties are inherent. Through the fuzzy `mssom`, we would obtain an approximation for each pixel that is easy to compute, yet always valid. This approximation could then be further improved by several means, forming a new, computationally efficient spectral unmixing algorithm.

Another aspect of spectral unmixing is the visualization and refinement of its results. Labitzke et al. propose an interactive method for improving spectral unmixing results, where the user can add or remove endmembers in each iteration [Labi 13a]. It would be worthwhile to integrate this algorithm with the visualization capabilities proposed in this thesis. Spectral distribution plots can be employed, whereas the transparency of spectral vectors under display would correspond to mixing weights, or constituents encoded in color, so that the color of a spectrum represents the mixture. The enhanced visualization capabilities should improve the capability of the user to find errors in the unmixing or endmember selection and correct them.

Chapter 7

Summary

Imaging spectroscopy allows acquisition of rich reflectance information that is not available with traditional RGB cameras and invisible to the human eye. Multispectral and hyperspectral images are well-established in various fields of application like remote sensing, astronomy, and microscopic spectroscopy. More recently, the availability of new sensor designs, more powerful processors and high-capacity storage further opened this imaging modality to a wider array of applications in medical diagnosis, food quality and safety control, and cultural heritage, to name a few.

In this thesis, we follow a universal approach on multispectral and hyperspectral image analysis. Instead of attending to a specific application, we work on fundamental problems and procedures to achieve a broad applicability of multispectral image data across applications. We tackle the question of how a computer system may guide a user towards a full understanding of the information contained in an image, independent of the capture and application scenario. Towards this goal, interactivity in the analysis is one of the most important, yet little researched aspects, which includes interactive time constraints for the employed algorithms. The generality of our approach makes it more difficult to evaluate proposed algorithms due to limited quantitative measures, lacking a specific scenario where the correct answer is already known or can be assessed by domain experts. On the other hand, the proposed generic solutions can be tuned to a wide array of emerging applications.

One of the findings of this work is that the self-organizing map (SOM) can function as such a generic tool, taking part in providing solutions for several areas of analysis and visualization. In Chapter 3, we discuss the importance of dimensionality reduction for many algorithms operating on high-dimensional data in general and hyperspectral image data specifically. Linear methods like the principal component analysis may fail to capture subtle differences in the data. One example is to distinguish differences in the reflectance curves of metamers, that are overshadowed by more general reflectance effects, e. g. the positioning of objects in relation to a light source and the resulting intensity of illumination. Nonlinear methods are often known to better represent the data manifold in a lower dimension. However, they are generally computationally expensive and not viable in an interactive setting. The SOM is an exception to the rule, as it allows for fast manifold-learning due to its robustness towards a smaller sample set and strong quantization of the learned data topology. We also propose a probabilitic manifold learning method that closely

resembles the SOM training, formulated as an expectation-maximization algorithm. Further, we show how a SOM can be trained more efficiently in terms of computational complexity. On a range of benchmark datasets and map configurations, our changes to the algorithm led to speed-up factors between 4.8 and 9.5. This allows training on an unseen image in interactive time. We also propose two extensions of the SOM to remedy its quantization effects and improve its representation of known data samples. One is based on using more rich information in the map lookup, through the collection of not one, but several best-matching units (BMU), and a ranking scheme to combine the information of these units. The other is a variation of the SOM learning process that combines input of labeled data with unlabeled data. Our experiments underline the SOM's robustness and versatility when applied to multispectral and hyperspectral data. Furthermore, the proposed extensions of the SOM in this thesis make it an even more powerful tool for data analysis regarding its data representation quality. This is shown in a classification scenario on a commonly used remote sensing benchmark image. There, the use of an efficiently trained larger map, our semi-supervised training method, and our modified map lookup greatly increase the classification performance both individually and when combined. On the widely used Indian Pines benchmark, overall accuracy was improved from a traditional SOM at 75.1 % to 84.3 %.

With the self-organizing map at our disposal, we then attend to three prominent fields of image processing that are relevant in multispectral and hyperspectral image analysis in Chapter 4, namely edge detection, clustering, and supervised segmentation. These are fundamental tools for automated and manual scene understanding.

Edge detection on multispectral and hyperspectral images is an issue short of a comprehensive solution. Previous work on R-ordering was an important step towards better reflecting the high-dimensional characteristics of the data when compared to gradient-based methods. However, as is seen in our experiments, the most prominent recent R-ordering method, RCMG, depends heavily on a well-performing dissimilarity measure. Toivanen et al. proposed a method based on dimensionality reduction via the self-organizing map. Unfortunately, their algorithm suffered from its ties to global pixel ordering, resulting in linearization artifacts. In this work, we introduce two pseudometrics that are also based on the topological learning of a SOM. These pseudometrics are, in contrast to established measures, data-driven. Our methods omit linearization and use the SOM more efficiently for edge detection while also retaining greater flexibility, without a disadvantage in computational performance. The first proposed pseudometric, SOM_1, is based on a traditional SOM readout. The SOM_2 pseudometric instead employs a multi-BMU lookup and the earth mover's distance for higher consistency and quality in the edge map. Our experimental evaluation reveals that both newly introduced measures, but most particularly SOM_2, significantly improve on the performance of both gradient-based and R-ordering-based edge detection for multispectral and hyperspectral images.

Unsupervised clustering of image data is an equally important image processing tool, which we treat in Section 4.2. Due to the rich pixel vectors of a multispectral image, mode-seeking clustering algorithms can provide very helpful segmentations without prior knowledge. We demonstrate this with the SG-FAMS method which is based on the well-known mean shift algorithm. However, the method is com-

putationally demanding and not suitable for an interactive analysis setting in the foreseeable future. Therefore, we propose two means of reducing the problem complexity. One, by employing superpixel pre-segmentation for a spatial sparsification of the data. Two, by relying on the vector quantization capabilities of the SOM to operate on a sparse representation of the sample distribution. We also investigate how the topology learned by a SOM can be directly employed for scene clustering. Our experiments show that superpixels are an effective tool not only on grayscale and RGB data, but also in the multispectral domain. By combining superpixels with SG-FAMS, we obtain a fast, yet reliable unsupervised clustering method. In cases where spatial resolution needs to be fully preserved, a likewise fast clustering can be performed with the SOM. We related the output of the SOM-based SG-FAMS method with the original SG-FAMS in terms of agreement between segmentation results and found an average Rand index of 0.93 on our test dataset. On the other hand, our proposed topology-based method, which only relies on the relationship of model vectors with each other both in feature space and SOM topology, is also shown to be a viable tool for obtaining an especially fast, high quality segmentation, with an achieved average computation time on our test dataset of 3.4 seconds as compared to over nine minutes for SG-FAMS. The SOM based methods are extended for fuzzy clustering using our proposed rank-weighted multi-BMU lookup scheme. Fuzzy clustering is useful in areas where a hard assignment is unsuitable to convey mixed information contained in a pixel, as demonstrated by an example.

Tending to our goal of a universal framework for interactive image analysis, we also explore the viability of seed-based local segmentation in multispectral images in Section 4.3. We adapt the power watershed framework, which combines several well-established graph-based supervised segmentation methods, by incorporating similarity measures known from the field of spectral matching as well as our data-driven approach previously developed for edge detection. To test algorithmic performance, we design a benchmark that covers a wider range of segmentation tasks on a publicly available multispectral dataset. Our results show that a straightforward attempt of using the Chebyshev distance does not yield satisfactory results with an average F_1-score of 0.66. Both proposed adaptations improve the segmentation performance significantly and achieve average F_1-scores between 0.90 and 0.93. As a result, we obtain a versatile supervised segmentation method that works without putting the burden on the user to carefully place a higher number of foreground and background seeds. The segmentation algorithm is shown to operate reliably and fast enough to enable quick refinements or further exploration.

Unsupervised and supervised analysis algorithms go hand-in-hand with a strong visualization when inspecting data. In Chapter 5 we discuss novel ways of visualizing multispectral and hyperspectral images. Existing analysis frameworks are limited to traditional display and plotting tools in their visualization capabilities, or provide additional visualizations that are very application-specific. A generally powerful concept is false color visualization, where specific information is encoded in color and displayed in the spatial layout of the image. Unsupervised false-coloring is strongly tied to dimensionality reduction. The most popular methods under this category are based on PCA. However, PCA often fails to reflect relevant information in the first three principal components used for false-coloring. In Section 5.2, we

propose a fast, non-linear false-color display that is based on the SOM. We note that when using a traditional SOM with a single-BMU lookup, a low-quality display of low entropy is obtained, suffering from the strong quantization of the map. Using our new rank-weighted multi-BMU lookup, entropy values of the resulting display channels greatly surpass the ones achieved by PCA, ranging between 0.88 and 0.96 on average for the two datasets we evaluated. We also find through visual comparison that spectral discrimination appears greatly improved when compared to PCA.

While plots of single spectra, or data ranges are common in existing analysis software, there exists no visualization of the whole spectral distribution contained in an image. We fill this gap with the help of the parallel coordinate visualization method, as introduced in Section 5.3. The distribution plots build on a sparse, high-dimensional histogram to combine similar spectra and draw them efficiently with varying opacity. Furthermore, spectra can be color-coded according to data labels or by employing a false-coloring method. In combination with different data descriptors, the user can quickly gain an idea about the relationship of objects in the scene in the high-dimensional space. In our experiments, we show how the drawing of histogram-based spectral distributions can be done with very low effort, without significant loss in accuracy. We further discuss how the brushing and linking technique, where the user can manipulate the distribution plots and spatial displays of the data simultaneously, forms a concept for interactive exploration that is new to manual hyperspectral image analysis.

A software framework coined Gerbil was developed that combines the proposed new approaches to both automated analysis tools and visualization techniques with established tools. It demonstrates a new workflow in multispectral and hyperspectral image inspection and exploration. Gerbil is developed by an open-source software project in the tradition of free software frameworks in the signal processing research community such as OpenCV or Weka. The project received continued support from the European Space Agency in the years 2012–2016 as well as individual contributions. It is in use by several research groups and industry worldwide. See Appendix B on page 163 for a current feature overview of the software. One purpose of this work and the Gerbil software project is to help broaden the application field of imaging spectroscopy. Domain experts who did not work with multispectral data before need assistance in understanding and utilizing the modality. We believe that an intuitive, responsive graphical user interface that encapsulates generally applicable algorithms can significantly facilitate the entry to multispectral image analysis.

The software originating from this work serves as an example on how intuitive exploration of multispectral and hyperspectral image data from a broad variety of application domains is possible, by adapting established methods from traditional monochromatic and trichromatic computer vision research as well as non-linear manifold learning methods and combining them with powerful, interactive data visualization concepts.

Appendix A

Symbols and Acronyms

Acronyms

Acronym	Meaning
AQE	Average quantization error
BMU	Best-matching unit(s)
FAMS	Fast-adaptive mean shift [Geor 03]
FH04	Algorithm by Felzenszwalb and Huttenlocher [Felz 04]
FSPMS	Full superpixel mean shift
HSV	Hue-Saturation-Value color space
ISOMAP	Optimal isometric mapping [Tene 00]
LDA	Linear Discriminant Analysis
MQE	Maximum quantization error
PCA	Principal component analysis
PDF	Probability density function
PSPMS	Per-superpixel mean shift
RCMG	Robust color morphological gradient [Evan 06]
RGB	Red-Green-Blue color space
SA(M)	Spectral angle (mapper)
SEDMI	Saliency-based edge detection in multispectral images [Dinh 11]
SG-FAMS	FAMS on the spectral gradient
SID	Spectral information divergence
SOM	Self-organizing map [Koho 01]
sRGB	standard RGB color space [Stok 96]
SVM	Support vector machine

Symbols

In general, a set $\mathcal{A}$ consists of either scalars a, or vectors $\boldsymbol{a}$ with coefficients a_d. The size of $\mathcal{A}$ is denoted as $n_A = |\mathcal{A}|$. A matrix is denoted as $\boldsymbol{A}$. Chapter-specific symbols are listed after symbols that are used throughout the thesis.

Sets

Set	Element	Count	Meaning
	$c, \boldsymbol{c}$	n_C	best-matching unit index / indices
$\mathcal{K}$	k		reference labels
$\mathcal{L}$	l	n_L	labels, class assignments
$\mathcal{M}$	$\boldsymbol{m}$	n_M	SOM model vectors
$\mathcal{M}$	$\boldsymbol{\mu}$		cluster centers
$\mathcal{R}$	$\boldsymbol{r}, r$		locations in a map
$\mathcal{S}$		n_S	samples
$\mathcal{X}$	$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$	n_X	pixels

Other counts not listed above

Symbol	Meaning
n_D	number of bands / pixel dimensionality
n_K	parameter k for kNN, k-means, FAMS bandwidth selection
n'_M	sidelength of SOM topology
n_R	dimensionality of SOM topology

Indices

Symbol	Meaning
d, e	band/coefficient in $[1, n_D]$
i, j	general index
i	spectral vector, typically in $[1, n_X]$
j	spectral vector, typically in $[1, n_K]$
k, l	SOM unit
l	x-coordinate
m	y-coordinate
s	sample / iteration

Functions

Symbol	Meaning
$\alpha(s)$	SOM learning-rate factor
COV($\boldsymbol{x}, \boldsymbol{y}$)	covariance
d($\boldsymbol{x}, \boldsymbol{y}$)	a distance function
e(λ)	spectrum of light source
E($\boldsymbol{p}$)	relative direction of light source
I($\boldsymbol{p}, \lambda$), I(l, m)	sensor response
$\kappa(\boldsymbol{x})$	a kernel
$\mathrm{L}_\lambda(\boldsymbol{p}, \lambda)$	spectral gradient
N($\boldsymbol{x}$)	Normalization in L_2
NED($\boldsymbol{x}, \boldsymbol{y}$)	normalized Euclidean distance
R($\boldsymbol{p}, \lambda$)	relative amount of reflected light
S($\boldsymbol{p}, \lambda$)	reflectance function
$\sigma(s)$	SOM kernel width
SA($\boldsymbol{x}, \boldsymbol{y}$)	spectral angle
SCM($\boldsymbol{x}, \boldsymbol{y}$)	spectral correlation mapper
SID($\boldsymbol{x}, \boldsymbol{y}$)	spectral information divergence
$\mathrm{SIDSAM}_{1,2}(\boldsymbol{x}, \boldsymbol{y})$	combined SA, SID measure
sRGB($\boldsymbol{x}$)	conversion to sRGB
VAR($\boldsymbol{x}$)	variance

Other symbols

Symbol	Meaning
$\boldsymbol{b}$	a spectral band
λ	wavelength
$\boldsymbol{p}$	point in a scene
ρ	sample Pearson correlation coefficient
$\boldsymbol{r}'$	non-discrete representative location
$\boldsymbol{w}, w$	vector of weights, single weight

Chapter 3

Symbol	Meaning
$\boldsymbol{A}$	ICA mixing matrix
$\mathrm{a}(\boldsymbol{x})$	class response by classifier
α	mixture weight
$\boldsymbol{C}$	graph clique set
η	a prior
$G_M = (\mathcal{X}, \mathcal{E}_M)$	graph on the manifold M
$G_Q = (\mathcal{R}, \mathcal{E}_Q)$	graph on the manifold Q
$\mathrm{h}(\cdot, \cdot)$	neighborhood function
M	manifold in the input feature space
n_P	number of principal components
Q	manifold in a low-dimensional space
$\boldsymbol{S}$	covariance matrix
$\boldsymbol{s}$	ICA signal sources
T	temperature of Gibbs distribution
$\boldsymbol{u}$	PCA projection vector
$\boldsymbol{v}_l$	votes for a class
$\omega_l(\cdot)$	weighting function in supervised learning
$\Xi, \boldsymbol{\xi}$	Set of parameter vectors, a single vector
Z_Q	partition function of Gibbs field

Chapter 4

Symbol	Meaning
$\mathcal{B}$	set of background seeds
c	parameter to FH04
C	compactness measure
$\mathcal{E}, e$	graph edges, single edge
$\mathcal{F}$	set of foreground seeds
$\gamma(\boldsymbol{x})$	a gradient kernel, κ shadow of γ
$\boldsymbol{G}_H, \boldsymbol{G}_V$	horizontal and vertical edge maps
$\mathrm{G_H}(l, m)$	x-directional gradient function
$\mathrm{G_V}(l, m)$	y-directional gradient function
h, h_i	(per-sample) bandwidth in mean shift / kernel density estimation
$\boldsymbol{L}$	omnidirectional Laplacian filter matrix
$\mathrm{L}(l, m)$	omnidirectional Laplacian filter
$\boldsymbol{m}$	mean shift vector
m	parameter to FH04
n_S	number of superpixels
n_Y	amount of pixels represented by a SOM unit
$\boldsymbol{p}$	power watershed segmentation result
p	pre-factor for FAMS bandwidth selection
p, q	parameters for power watersheds
$\boldsymbol{R}$	location map (index for each pixel)
$\mathcal{R}_{\boldsymbol{x}}$	set of BMU locations for a pixel
R	Rand index
$\boldsymbol{S}$	Sobel filter matrix
$\mathcal{S}, \mathcal{S}_j$	a set of pixels / a superpixel
$\boldsymbol{s}_j$	centroid of a superpixel
$\mathcal{V}, v$	graph vertices, single vertex
w_F, w_B	weights for foreground and background affiliation

Chapter 5

Symbol	Meaning
α	opacity of a bin/polyline
$\gamma(x)$	sRGB gamma correction function
$\mathrm{H}(x)$	entropy function
h	calculated entropy
n_B	number of histogram bins per dimension

Appendix B

Software Framework Features

Figure B.1 shows the modular architecture of Gerbil.

Input/Output

- Support of a wide range of image formats, including ENVI, FITS, ESRI, TIFF
- Custom file format, easy data import/export with Matlab
- Processing of image data that does not fit into system memory
- Reading and writing of label files, writing of spectral plots

Data Processing

- Data reduction:

  1. Region-of-interest selection, spatial masking
  2. Band range selection
  3. Interpolation, binning in the spectral domain

- Normalization:

  1. Data range calculation and histogramming
  2. Automatic, manual data normalization and range clamping
  3. Illuminant exchange

- Preprocessing:

  1. Flipping/transposition/rotation
  2. Apply logarithm, L2 normalization
  3. Gaussian kernel denoising

- Feature extraction:

  1. Spectral gradient
  2. Principal Component Analysis (PCA)
  3. Band-wise correlation
  4. Manhattan, Euclidean, Chebyshev norm
  5. Spectral Angle Mapper, Spectral Information Divergence, SIDSAM
  6. Earth mover's distance and statistical measures

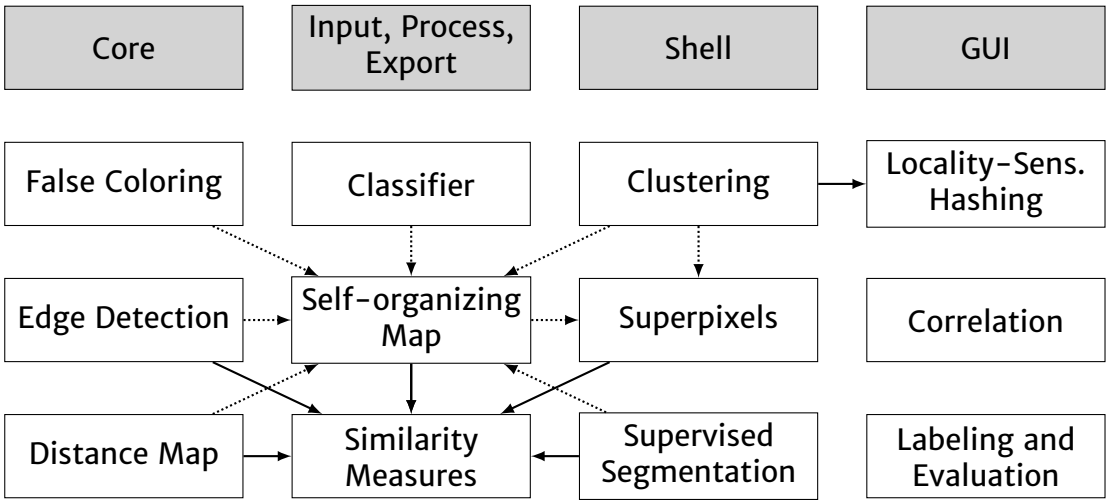


Figure B.1: Software modules. Solid arrows denote dependencies, dotted arrows denote optional dependencies.

- Label manipulation:

  1. Find labels via supervised segmentation
  2. Create and manipulate labels via thresholding
  3. Edit labels with various label brushes
  4. Merge, delete etc. labels in a separate view

Visualization

- Spectral Plots:

  1. Interactive Parallel-coordinates based spectral plots
  2. Display distribution in feature spaces including:
     (a) Original spectra
     (b) L_2-normalized spectra
     (c) Spectral gradient
     (d) Principal Component Analysis
  3. Color-coding based on labels or true color/false color
  4. Configurable drawing quality

- Spatial Plots:

  1. Display of image bands, spectral gradient bands, PCA components with translucent label overlay
  2. True-color display via CIE XYZ
  3. False-color display via PCA, Self-organizing Map (SOM)
  4. Distance Map for reference point/region

- Zoom/pan in all views, export views to file

Algorithms

- Dimensionality reduction via PCA, SOM
- Edge detection:
    1. Hyperspectral Laplace operator
    2. Robust color morphological gradient (RCMG)
    3. Canny edge detector
    4. SOM-based ordering
    5. SOM pseudometric for Laplace, RCMG, Canny
- Segmentation:
    1. Supervised segmentation via Graph Cuts, Power Watersheds
    2. Superpixel segmentation
- Global clustering:
    1. k-Means++
    2. Fast-adaptive mean shift (FAMS)
    3. Accelerated FAMS via superpixels
    4. Accelerated FANS via SOM pre-processing
    5. Topological SOM
- Classification:
    1. k-Nearest Neighbor
    2. Semi-supervised SOM
    3. Cross-validation framework

Interface

- Hardware-accelerated GUI on all major desktop platforms
- Command-line interface for batch processing and automated evaluation
- C++-API for image data access and incorporation of new methods
- Modularized build, command and parameter handling

Appendix C

Self-organizing Map Configurations

We provide one parameter set for each combination of size and topology. When size and topology are given in the text, the corresponding parameter set was used in testing. The default parameters across all configurations, if not specified otherwise, are:

- $s_{\max} = 50\,000$
- $\sigma(s_{\max}) = 1$
- $\alpha(1) = 0.75$
- $\alpha(s_{\max}) = 0.01$

Square

- $n'_M = 16, \sigma(1) = 6$
- $n'_M = 24, \sigma(1) = 8$
- $n'_M = 32, \sigma(1) = 12$
- $n'_M = 48, \sigma(1) = 16$
- $n'_M = 64, \sigma(1) = 24$

Cube

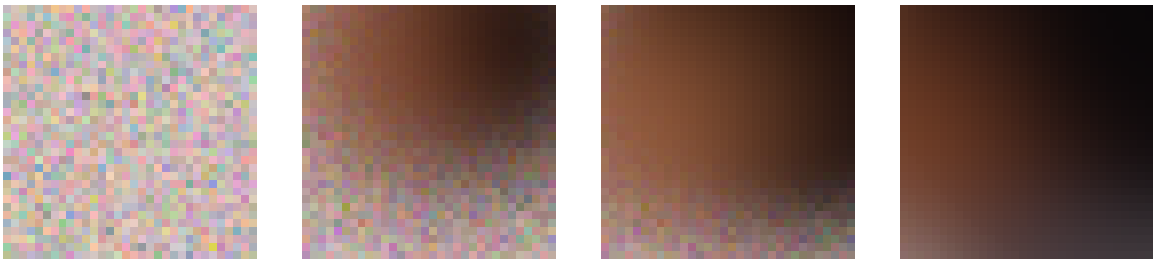
- $n'_M = 6, \sigma(1) = 2, \sigma(s_{\max}) = 0.5$
- $n'_M = 8, \sigma(1) = 3, \sigma(s_{\max}) = 0.75$
- $n'_M = 10, \sigma(1) = 4$
- $n'_M = 13, \sigma(1) = 5$
- $n'_M = 15, \sigma(1) = 5.5$
- $n'_M = 16, \sigma(1) = 6$
- $n'_M = 20, \sigma(1) = 12$

Tesseract

- $n'_M = 4, \sigma(1) = 1, \sigma(s_{\max}) = 0.3$
- $n'_M = 5, \sigma(1) = 1.5, \sigma(s_{\max}) = 0.4$
- $n'_M = 6, \sigma(1) = 2, \sigma(s_{\max}) = 0.5$
- $n'_M = 7, \sigma(1) = 2.5, \sigma(s_{\max}) = 0.6$
- $n'_M = 8, \sigma(1) = 3, \sigma(s_{\max}) = 0.7$

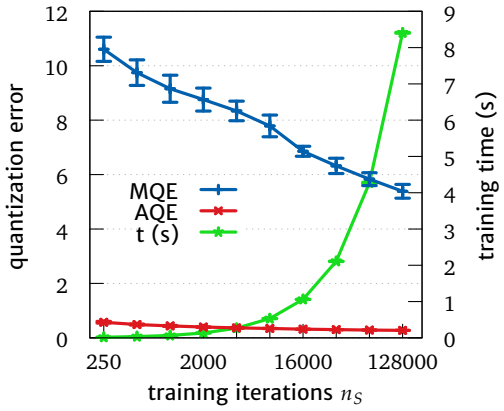
Appendix D

Experimental results

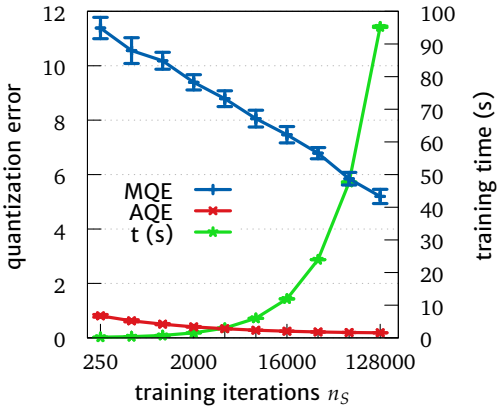


(a) initialization **(b)** iteration 5 **(c)** iteration 15 **(d)** iteration 50

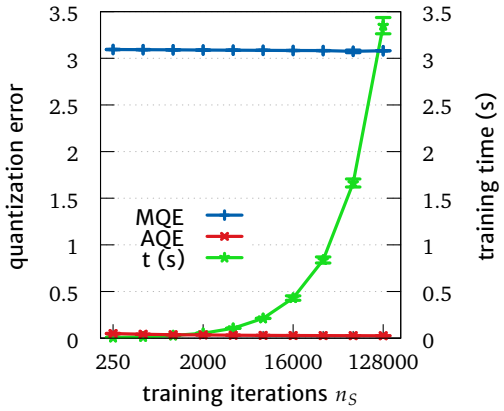
Figure D.1: Visualization of SOM training progress on *Egyptian Statue* image at the very beginning of the training. See Figure 3.3 on page 35 for later stages. Maps are rendered in true color.



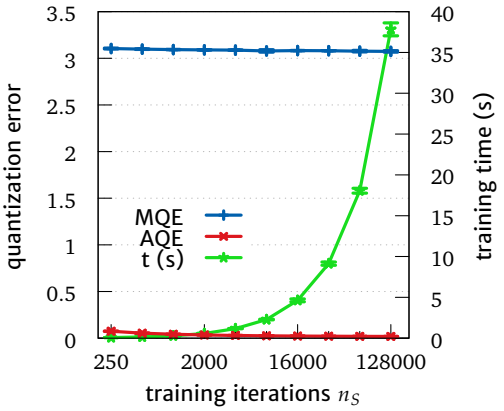
(a) *Dürer 1*, square, $n_M = 256$



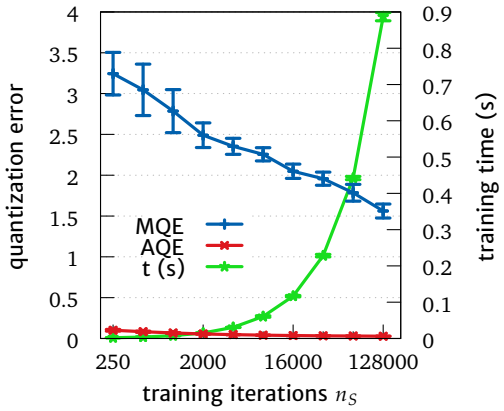
(b) *Dürer 1*, cube, $n_M = 2197$



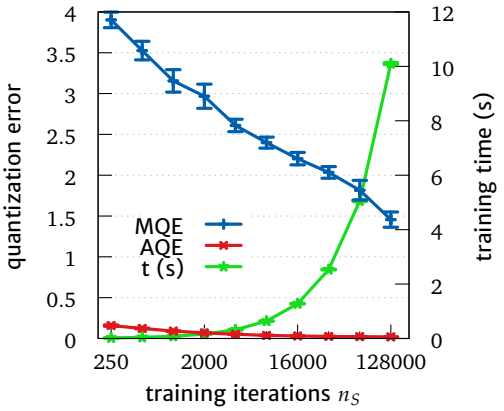
(c) *D.C. Mall*, square, $n_M = 256$



(d) *D.C. Mall*, cube, $n_M = 2197$

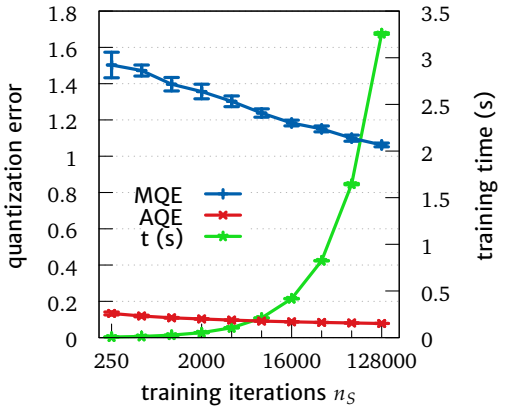


(e) *Fake and Real Peppers*, square, $n_M = 256$

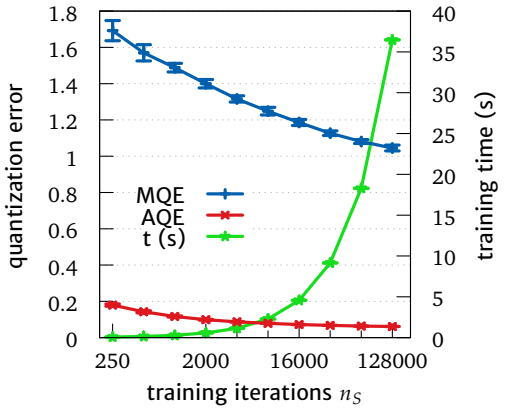


(f) *Fake and Real Peppers*, cube, $n_M = 2197$

Figure D.2: Quantization error for different SOM parameters. n_S is plotted on a logarithmic scale. See also Figure D.3.

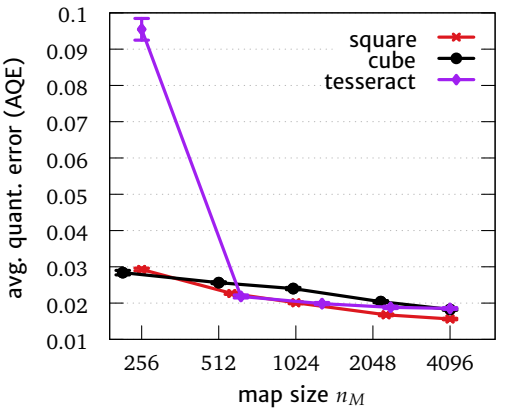


(a) *Indian Pines*, square, $n_M = 256$

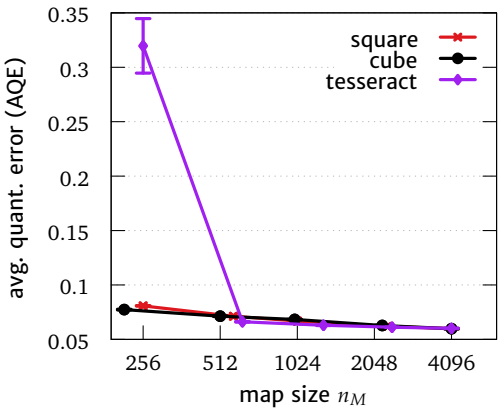


(b) *Indian Pines*, cube, $n_M = 2197$

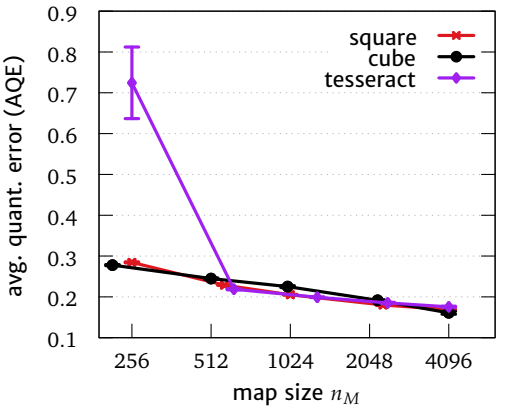
Figure D.3: Quantization error for different SOM parameters *(cont.)*. n_S is plotted on a logarithmic scale.



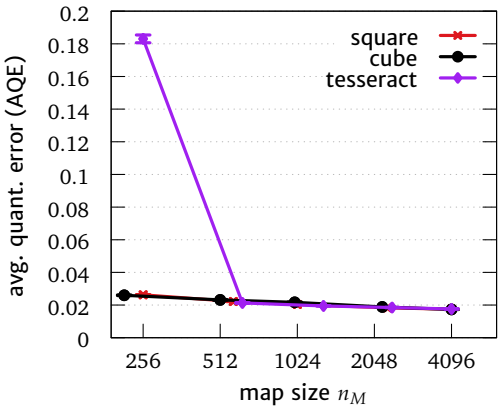
(a) *Fake and Real Peppers*



(b) *Indian Pines*



(c) *Dürer 1*



(d) *D.C. Mall*

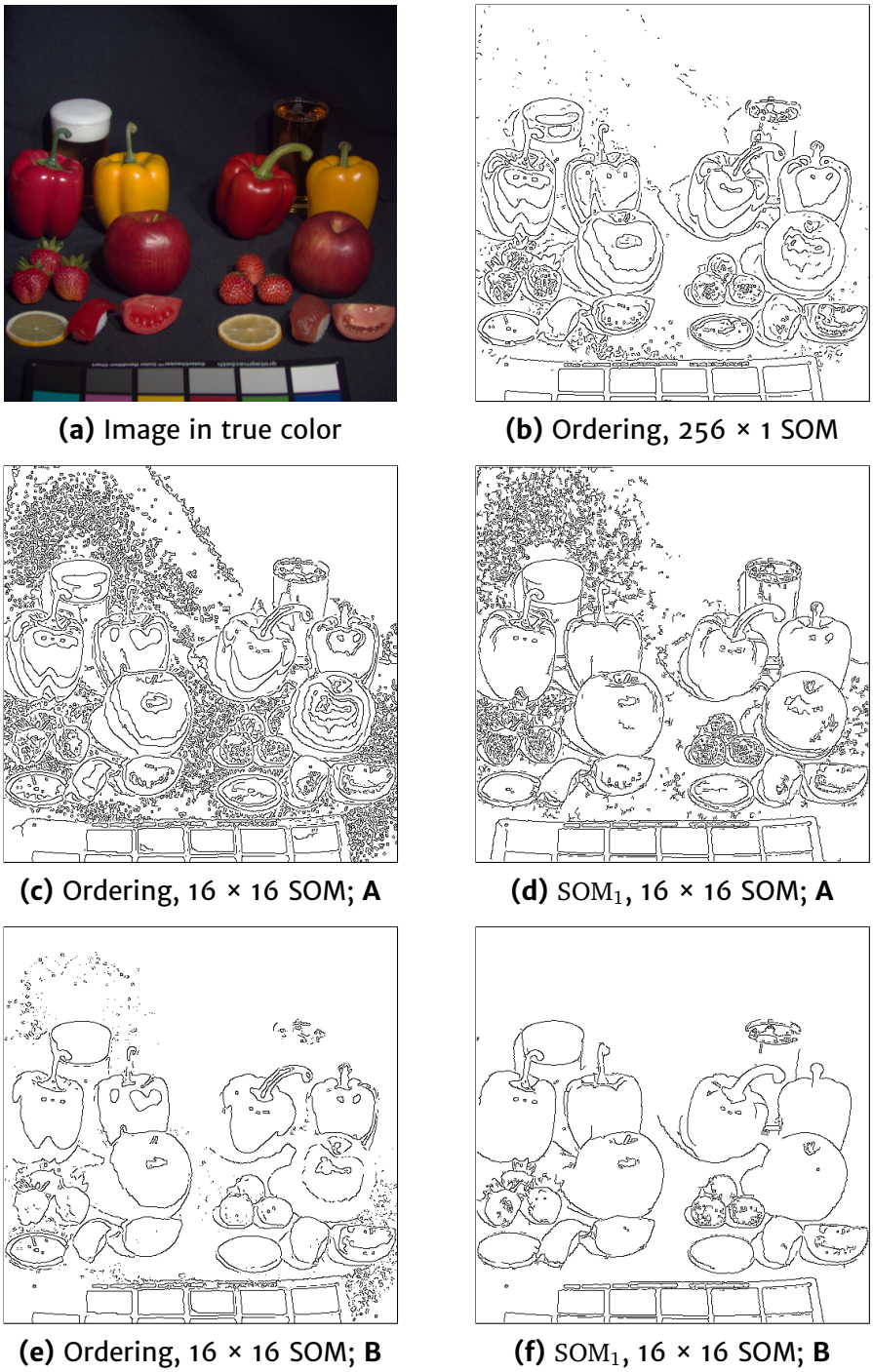
Figure D.4: Quantization error for different SOM shapes and sizes. n_M is plotted on a logarithmic scale.

Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	#samples
1	**0.939**		0.002	0.002	0.002	0.018	0.008	0.008	0.002		0.004	0.002		0.010			483
2	0.150	**0.500**		0.050				0.250								0.050	20
3	0.006		**0.609**	0.173			0.027		0.049		0.002	0.116		0.014			830
4	0.005	0.001	0.030	**0.807**			0.050	0.002	0.015		0.001	0.077		0.006			2455
5	0.021				**0.717**				0.021						0.239		46
6	0.033					**0.939**		0.025									1265
7	0.008	0.001	0.033	0.182			**0.707**		0.018		0.004	0.040		0.004			972
8	0.240	0.015	0.002	0.005	0.002	0.209		**0.432**	0.005		0.069					0.015	386
9	0.010		0.080	0.106	0.001		0.077	0.005	**0.483**		0.001	0.207		0.025			593
10	0.035									**0.928**					0.035		28
11	0.030			0.001		0.002	0.001	0.041			**0.916**			0.006			730
12	0.004		0.059	0.199			0.050		0.060		0.002	**0.602**		0.020			1428
13	0.021			0.043								0.021	**0.913**				93
14	0.029		0.021	0.160			0.084	0.012	0.046		0.054	0.232		**0.358**			237
15	0.004			0.004	0.014					0.004					**0.972**		478
16	0.024	0.019				0.004		0.019								**0.931**	205

Table D.1: Confusion matrix for baseline method with SOM of $n_M = 1296$, unsupervised training, and single BMU lookup. Overall accuracy: 75.1 %, average accuracy: 73.5 %.

Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	#samples
1	**0.960**			0.004		0.006	0.004	0.006	0.008		0.004	0.004		0.002			483
2	0.100	**0.650**					0.100	0.100				0.050					20
3			**0.721**	0.138			0.021	0.001	0.032			0.072		0.012			830
4	0.001		0.028	**0.841**			0.053	0.002	0.011			0.056		0.004			2455
5	0.195				**0.695**										0.108		46
6	0.006					**0.972**		0.019									1265
7	0.001	0.001	0.016	0.125			**0.806**	0.001	0.012		0.004	0.028		0.003			972
8	0.012	0.002	0.002	0.015		0.186	0.007	**0.681**	0.007		0.067			0.002		0.012	386
9	0.001		0.021	0.064			0.052	0.001	**0.758**		0.003	0.075	0.003	0.016			593
10	0.357									**0.571**					0.071		28
11	0.001			0.001		0.001		0.013			**0.982**						730
12			0.033	0.121			0.050		0.022		0.002	**0.756**		0.012			1428
13									0.021			0.021	**0.956**				93
14			0.054	0.059			0.042		0.012		0.004	0.143		**0.679**	0.004		237
15					0.004										**0.995**		478
16	0.004						0.004					0.004				**0.985**	205

Table D.2: Confusion matrix for best-performing method with SOM of $n_M = 4096$, semi-supervised training, and ranked BMU lookup of $n_C = 15$. Overall accuracy: 84.3 %, average accuracy: 81.3 %.



(a) Image in true color

(b) Ordering, 256 × 1 SOM

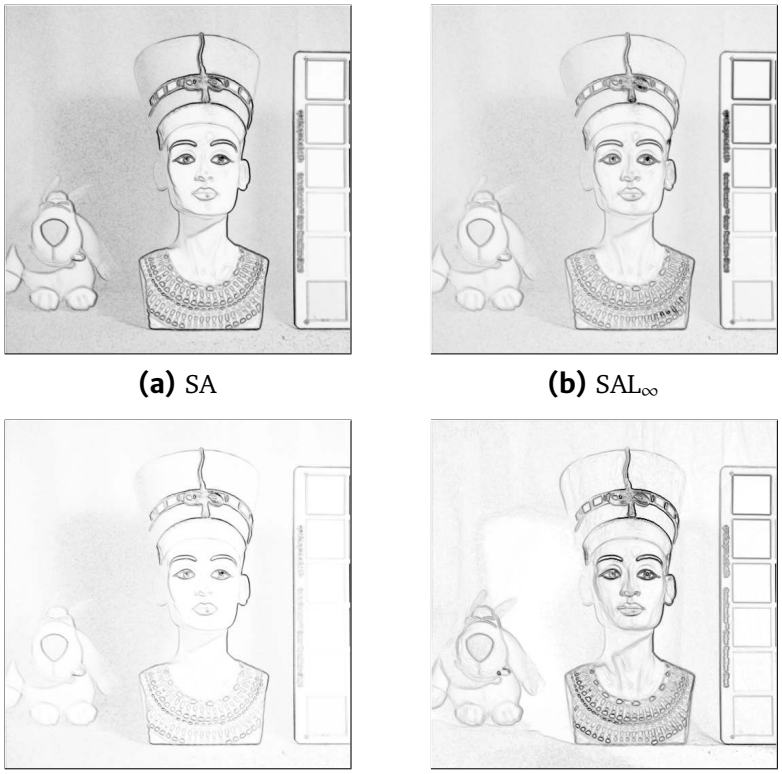
(c) Ordering, 16 × 16 SOM; **A**

(d) SOM_1, 16 × 16 SOM; **A**

(e) Ordering, 16 × 16 SOM; **B**

(f) SOM_1, 16 × 16 SOM; **B**

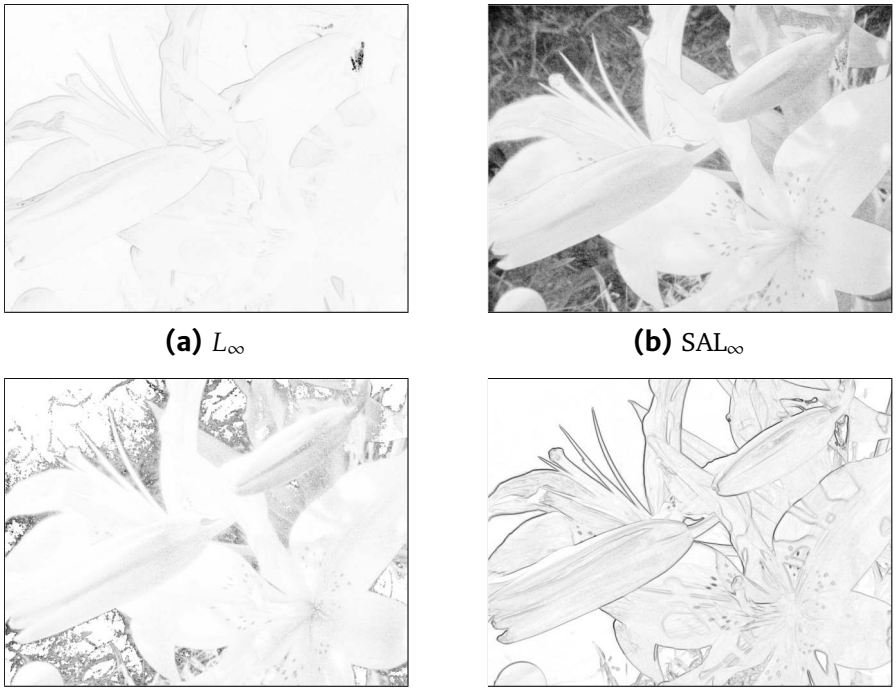
Figure D.5: Canny edge detector results on *Fake and Real Food* image. Canny parameter criteria: **A**) best object contour preservation; **B**) minimum fine-grained noise introduced by object/background texture.



(a) SA **(b)** SAL_∞

(c) SID **(d)** SOM_2

Figure D.6: RCMG edge detector results on *Egyptian Statue* image.



(a) L_∞ **(b)** SAL_∞

(c) SID **(d)** SOM_2

Figure D.7: RCMG edge detector results on *Cyflower* image.

Cloth

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		14			SG-FAMS*		−1	+2.3	0.97
PSPMS	0.05	+3	+5.4	0.94	`mssom`	8000	+5	+0.3	0.84
	0.25	+4	+12.7	0.91		1000	−3	+4.3	0.92
FSPMS	0.05	+0	+4.6	0.92	`somtopo`	4096	+1	+3.2	0.86
	0.25	+0	+12.1	0.87		1024	+1	+7.2	0.82

Egyptian Statue

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		13			SG-FAMS*		+3	+8.6	0.90
PSPMS	0.05	+87	−0.7	0.89	`mssom`	8000	+11	−28.9	0.90
	0.25	+64	+3.4	0.87		1000	−1	−21.6	0.83
FSPMS	0.05	+22	−4.9	0.87	`somtopo`	4096	+4	−29.2	0.86
	0.25	+14	−14.9	0.85		1024	+3	−30.6	0.85

Fake and Real Peppers

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		19			SG-FAMS*		+0	−0.9	0.99
PSPMS	0.05	+6	−2.2	0.97	`mssom`	8000	+2	−1.3	0.97
	0.25	+4	+1.7	0.97		1000	−6	+4.8	0.95
FSPMS	0.05	+9	−1.4	0.96	`somtopo`	4096	−6	+8.5	0.94
	0.25	+2	+7.5	0.95		1024	−5	+3.6	0.95

Fake and Real Food

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		26			SG-FAMS*		+0	−1.8	0.98
PSPMS	0.05	+6	−2.9	0.98	`mssom`	8000	−1	−1.1	0.94
	0.25	+6	0.0	0.97		1000	−5	−0.1	0.93
FSPMS	0.05	+1	−3.2	0.96	`somtopo`	4096	−9	+6.5	0.90
	0.25	−2	+2.8	0.92		1024	−7	+1.9	0.93

Flowers

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		25			SG-FAMS*		−2	−18.3	0.98
PSPMS	0.05	+21	−8.9	0.97	`mssom`	8000	+0	−21.1	0.96
	0.25	+30	−5.9	0.97		1000	−7	−14.1	0.95
FSPMS	0.05	+0	−15.1	0.96	`somtopo`	4096	−7	−16.0	0.93
	0.25	−8	−14.1	0.96		1024	−11	−12.4	0.94

Table D.3: Statistical results on individual images. Listed under # is number or segments, or difference therein. ΔC is given in percent.

Pompoms

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		25			SG-FAMS*		−1	+1.0	0.98
PSPMS	0.05	+8	−0.7	0.98	mssom	8000	−6	0.0	0.94
	0.25	+7	+2.4	0.97		1000	−13	+4.6	0.94
FSPMS	0.05	−3	−0.6	0.96	somtopo	4096	−8	+2.0	0.93
	0.25	−8	+8.4	0.94		1024	−11	+6.5	0.92

Superballs

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		18			SG-FAMS*		−1	+0.9	0.99
PSPMS	0.05	+3	−0.1	0.98	mssom	8000	+4	−3.4	0.95
	0.25	+4	+0.6	0.98		1000	−4	+5.0	0.93
FSPMS	0.05	−1	+2.2	0.95	somtopo	4096	−2	+1.3	0.93
	0.25	−1	+5.5	0.94		1024	−4	+2.4	0.93

Thread Spools

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		32			SG-FAMS*		+0	+0.2	0.99
PSPMS	0.05	+4	0.0	0.99	mssom	8000	+0	−0.9	0.97
	0.25	+8	+0.4	0.98		1000	−15	+9.7	0.93
FSPMS	0.05	+1	−4.7	0.97	somtopo	4096	−14	+13.4	0.93
	0.25	−15	+14.3	0.91		1024	−17	+12.1	0.92

Watercolors

Algorithm	c	#	ΔC	R	Algorithm	n_M	#	ΔC	R
SG-FAMS		32			SG-FAMS*		−7	+3.2	0.89
PSPMS	0.05	+9	+0.5	0.94	mssom	8000	−3	+3.0	0.92
	0.25	+6	−0.2	0.95		1000	−20	+6.7	0.81
FSPMS	0.05	−7	+0.7	0.93	somtopo	4096	−20	+10.6	0.84
	0.25	−12	+12.3	0.61		1024	−19	+5.2	0.85

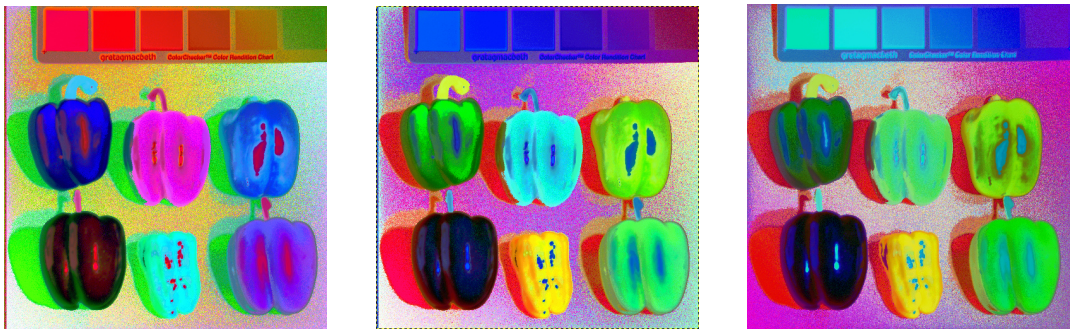
Table D.4: Statistical results on individual images. Listed under # is number or segments, or difference therein. ΔC is given in percent.

(a) SG-FAMS **(b)** FSPMS **(c)** `somtopo`, $n_M = 4096$

Figure D.8: Additional segmentation results on *Ribeira* image. For HRK and `mssom` see Figure 4.26 on page 106.

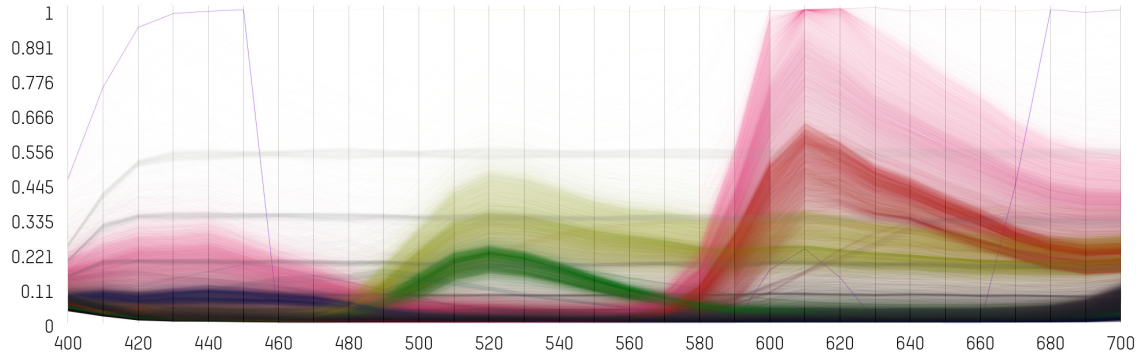
(a) true color **(b)** PCA **(c)** SOM

Figure D.9: Color displays of *Egyptian Statue* image in original feature space.

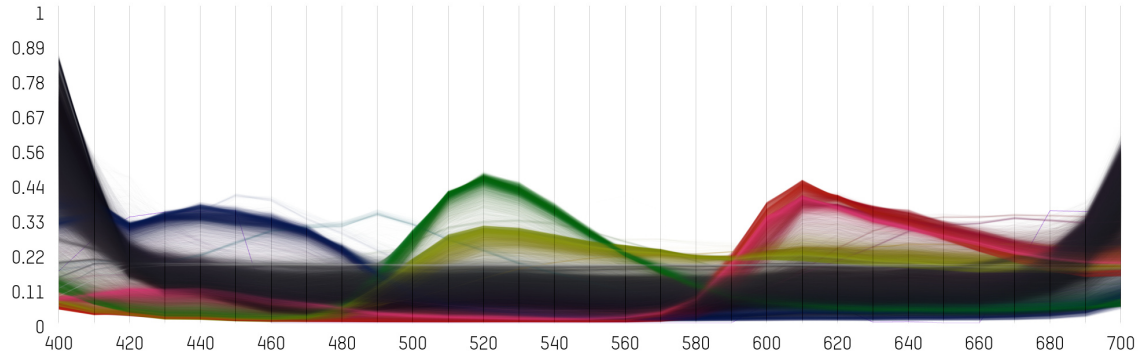


(a) user-provided **(b)** color-rotated **(c)** published [Jord 13a]

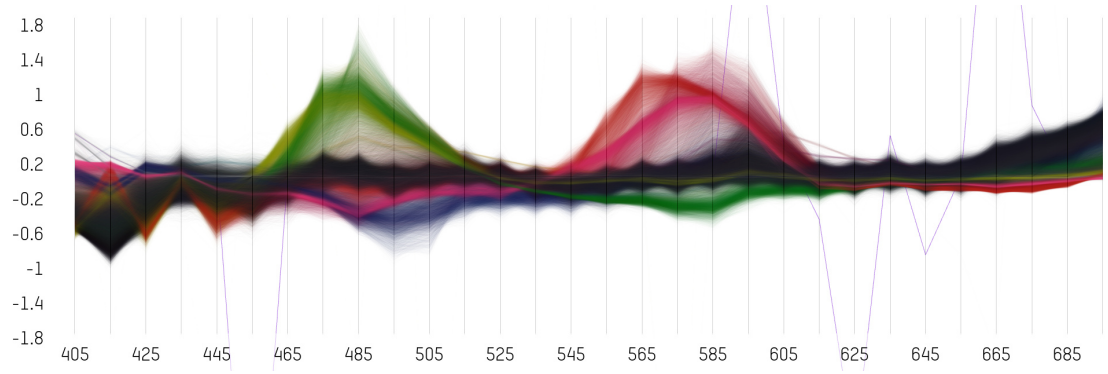
Figure D.10: SOM false-color displays of *Fake and Real Peppers* image computed on the spectral gradient. The result depicted in (b) was computed by a color-wheel rotation of (a) by 236.8°. This simple 2-D rotation explains most of the difference between (a) and (c).



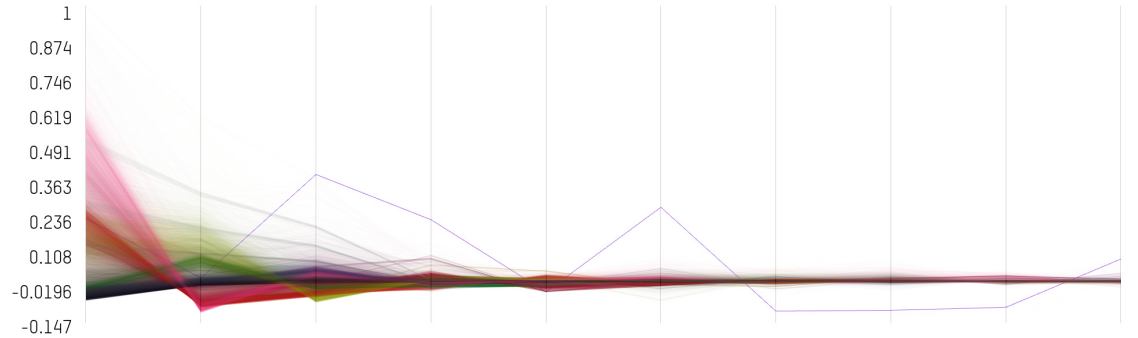
(a) spectral distribution



(b) normalized spectral distribution



(c) spectral gradient distribution



(d) principal component distribution

Figure D.11: Distribution plots of *Superballs* image. Spectra are colored in true color. The image is depicted in Figure 4.28a on page 109.

List of Figures

List of Tables

Bibliography

[Acha 12] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 11, pp. 2274–2282, 2012.

[Andr 99] D. Androutsos, K. N. Plataniotis, and A. N. Venetsanopoulos. "A novel vector-based approach to color image retrieval using a vector angular-based distance measure". *Computer Vision and Image Understanding*, Vol. 75, No. 1, pp. 46–58, 1999.

[Ange 00] E. Angelopoulou. "Objective Colour from Multispectral Imaging". In: *European Conference on Computer Vision*, pp. 359–374, Springer-Verlag, Dublin, June 2000.

[Ange 01] E. Angelopoulo, R. Molana, and K. Daniilidis. "Multispectral skin color modeling". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 635–642, IEEE, Kauai, HI, Dec. 2001.

[Ange 05] E. Angelopoulou, M. Paranjape, and T. Kothavade. "MultiSSA: MultiSpectral Scene Analysis Software". http://www5.cs.fau.de/fileadmin/Persons/AngelopoulouElli/MultiSSA, June 2005. Retrieved 2016-09-18.

[Ange 07] E. Angelopoulou. "Specular Highlight Detection Based on the Fresnel Reflection Coefficient". In: *IEEE International Conference on Computer Vision*, IEEE, Rio de Janeiro, Oct. 2007.

[Arbe 11] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. "Contour Detection and Hierarchical Image Segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 5, pp. 898–916, May 2011.

[Arth 07] D. Arthur and S. Vassilvitskii. "k-means++: The advantages of careful seeding". In: *Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035, Society for Industrial and Applied Mathematics, New Orleans, LA, Jan. 2007.

[Bac 05] F. Bação, V. Lobo, and M. Painho. "Self-organizing maps as substitutes for k-means clustering". In: *International Conference on Computational Science*, pp. 476–483, Springer, Atlanta, GA, May 2005.

[Bach 06] C. M. Bachmann, T. L. Ainsworth, and R. A. Fusina. "Improved manifold coordinate representations of large-scale hyperspectral scenes". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 10, pp. 2786–2803, 2006.

[Bajc 04] P. Bajcsy and P. Groves. “Methodology for hyperspectral band selection”. *Photogrammetric Engineering & Remote Sensing*, Vol. 70, No. 7, pp. 793–802, 2004.

[Bakk 02] W. Bakker and K. Schmidt. “Hyperspectral edge filtering for measuring homogeneity of surface cover types”. *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 56, No. 4, pp. 246–256, 2002.

[Bald 09] A. Baldridge, S. Hook, C. Grove, and G. Rivera. “The ASTER spectral library version 2.0”. *Remote Sensing of Environment*, Vol. 113, No. 4, pp. 711–715, 2009.

[Ball 14] Ball Aerospace & Technologies Corp. “Opticks–Spectral Processing Extension”. https://opticks.org/display/opticksExt/Spectral+Processing, Sep. 2014. Retrieved 2016-10-09.

[Ball 16] Ball Aerospace & Technologies Corp. “Opticks”. http://opticks.org/, Feb. 2016. Retrieved 2016-10-09.

[Ball 65] G. H. Ball and D. J. Hall. “ISODATA, a novel method of data analysis and pattern classification”. Tech. Rep., Stanford Research Institute, Menlo Park, CA, Apr. 1965.

[Band 09] T. V. Bandos, L. Bruzzone, and G. Camps-Valls. “Classification of hyperspectral images with regularized linear discriminant analysis”. *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 47, No. 3, pp. 862–873, 2009.

[Barn 76] V. Barnett. “The ordering of multivariate data”. *Journal of the Royal Statistical Society. Series A (General)*, Vol. 139, No. 3, pp. 318–355, 1976.

[Baum 15] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe. “220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3”. https://purr.purdue.edu/publications/1947/1, Sep. 2015. Retrieved 2016-10-09.

[Beye 99] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. “When is ‘nearest neighbor’ meaningful?”. In: *International Conference on Database Theory*, pp. 217–235, Springer, Jerusalem, Jan. 1999.

[Bieh 02] L. Biehl and D. Landgrebe. “MultiSpec–a tool for multispectral-hyperspectral image data analysis”. *Computers & Geosciences*, Vol. 28, No. 10, pp. 1153–1159, 2002.

[Bieh 07] L. Biehl. “Eval-Ware: Hyperspectral Imaging [Best of the web]”. *IEEE Signal Processing Magazine*, Vol. 24, No. 4, pp. 125–126, 2007.

[Biou 13] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot. “Hyperspectral remote sensing data analysis and future challenges”. *IEEE Geoscience and Remote Sensing Magazine*, Vol. 1, No. 2, pp. 6–36, 2013.

[Bish 06] C. M. Bishop. *Pattern Recognition and Machine Learning*. *Information Science and Statistics*, Springer Science & Business Media, 2006.

[Bish 98] C. M. Bishop, M. Svensén, and C. K. I. Williams. “GTM: The Generative Topographic Mapping”. *Neural Computation*, Vol. 10, No. 1, pp. 215–234, 1998.

[Blic 95] T. Blickle and L. Thiele. "A Comparison of Selection Schemes used in Genetic Algorithms". In: *TIK Report*, 2nd Ed., Swiss Federal Institute of Technology ETH, Dec. 1995.

[Boar 06] J. Boardman, L. Biehl, R. Clark, F. Kruse, A. Mazer, and J. Torson. "Development and Implementation of Software Systems for Imaging Spectroscopy". In: *IEEE International Geoscience and Remote Sensing Symposium*, pp. 1969–1973, IEEE, Denver, CO, Aug. 2006.

[Boar 95] J. Boardman, F. Kruse, and R. Green. "Mapping target signatures via partial unmixing of AVIRIS data". In: *Summaries, Fifth JPL Airborne Earth Science Workshop*, Jet Propulsion Lab, California Institute of Technology, Pasadena, CA, Jan. 1995.

[Boyk 06] Y. Boykov and G. Funka-Lea. "Graph cuts and efficient ND image segmentation". *International Journal of Computer Vision*, Vol. 70, No. 2, pp. 109–131, 2006.

[Cai 07] S. Cai, Q. Du, and R. J. Moorhead. "Hyperspectral imagery visualization using double layers". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 10, pp. 3028–3036, 2007.

[Camp 05] G. Camps-Valls and L. Bruzzone. "Kernel-based methods for hyperspectral image classification". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 43, No. 6, pp. 1351–1362, 2005.

[Canc 06] L. C. Cancio, A. I. Batchinsky, J. R. Mansfield, S. Panasyuk, K. Hetz, D. Martini, B. S. Jordan, B. Tracey, and J. E. Freeman. "Hyperspectral imaging: a new approach to the diagnosis of hemorrhagic shock". *Journal of Trauma and Acute Care Surgery*, Vol. 60, No. 5, pp. 1087–1095, 2006.

[Cann 86] J. Canny. "A Computational Approach to Edge Detection". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, pp. 679–698, 1986.

[Casi 99] A. Casini, F. Lotti, M. Picollo, L. Stefani, and E. Buzzegoli. "Image spectroscopy mapping technique for noninvasive analysis of paintings". *Studies in conservation*, Vol. 44, No. 1, pp. 39–48, 1999.

[Chab 06] S. Chabrier, B. Emile, C. Rosenberger, and H. Laurent. "Unsupervised Performance Evaluation of Image Segmentation". *EURASIP Journal of Applied Signal Processing*, Vol. 2006, pp. 1–12, Jan. 2006.

[Chak 11] A. Chakrabarti and T. Zickler. "Statistics of Real-World Hyperspectral Images". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 193–200, IEEE, Colorado Springs, CO, June 2011.

[Chan 00] C. Chang. "An information-theoretic approach to spectral variability, similarity, and discrimination for hyperspectral image analysis". *IEEE Transactions on Information Theory*, Vol. 46, No. 5, pp. 1927–1932, 2000.

[Chan 99] C.-I. Chang, Q. Du, T.-L. Sun, and M. L. G. Althouse. "A joint band prioritization and band-decorrelation approach to band selection for hyperspectral image classification". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 37, No. 6, pp. 2631–2641, Nov. 1999.

[Cher 03] A. Cheriyadat and L. M. Bruce. "Why principal component analysis is not an appropriate feature extraction method for hyperspectral data". In: *IEEE International Geoscience and Remote Sensing Symposium*, pp. 3420–3422, IEEE, Toulouse, July 2003.

[Clar 07] R. N. Clark, G. A. Swayze, R. Wise, K. E. Livo, T. Hoefen, R. F. Kokaly, and S. J. Sutley. "USGS digital spectral library splib06a". *U.S. Geological Survey, Digital Data Series*, Vol. 231, 2007.

[Cola 06] P. Colantoni, R. Pillay, C. Lahanier, and D. Pitzalis. "Analysis of multispectral images of paintings". In: *European Signal Processing Conference*, European Association for Signal Processing, Florence, Sep. 2006.

[Coma 02] D. Comaniciu and P. Meer. "Mean shift: A robust approach toward feature space analysis". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 5, pp. 603–619, 2002.

[Coma 03] D. Comaniciu. "An algorithm for data-driven bandwidth selection". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 2, pp. 281–288, 2003.

[Cook 82] R. L. Cook and K. E. Torrance. "A reflectance model for computer graphics". *ACM Transactions on Graphics*, Vol. 1, No. 1, pp. 7–24, 1982.

[Corc 10] P. Corcoran, A. Winstanley, and P. Mooney. "Segmentation performance evaluation for object-based remotely sensed image analysis". *International Journal of Remote Sensing*, Vol. 31, No. 3, pp. 617–645, 2010.

[Cott 16] M. Cottrell, M. Olteanu, F. Rossi, and N. Villa-Vialaneix. "Theoretical and applied aspects of the self-organizing maps". In: *International Workshop on Advances in Self-Organizing Maps and Learning Vector Quantization*, pp. 3–26, Springer, Houston, TX, Jan. 2016.

[Coup 09] C. Couprie, L. Grady, L. Najman, and H. Talbot. "Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest". In: *IEEE International Conference on Computer Vision*, pp. 731–738, IEEE, Kyoto, Sep. 2009.

[Coup 11] C. Couprie, L. Grady, L. Najman, and H. Talbot. "Power Watershed: A Unifying Graph-Based Optimization Framework". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 7, pp. 1384–1399, July 2011.

[Cous 09] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. "Watershed Cuts: Minimum Spanning Forests and the Drop of Water Principle". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 8, pp. 1362–1374, 2009.

[Cucc 11] C. Cucci, A. Casini, M. Picollo, M. Poggesi, and L. Stefani. "Open issues in hyperspectral imaging for diagnostics on paintings: when high-spectral and spatial resolution turns into data redundancy". In: *O3A: Optics for Arts, Architecture, and Archaeology III*, pp. 808408–808408–10, 2011.

[Cui 09] M. Cui, A. Razdan, J. Hu, and P. Wonka. "Interactive hyperspectral image visualization using convex optimization". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 47, No. 6, pp. 1673–1684, 2009.

[Cui 10] W. Cui and Y. Zhang. "Graph Based Multispectral High Resolution Image Segmentation". In: *International Conference on Multimedia Technology*, pp. 1–5, IEEE, Ningbo, Oct. 2010.

[Cuma 91] A. Cumani. "Edge detection in multispectral images". *CVGIP: Graphical Models and Image Processing*, Vol. 53, No. 1, pp. 40–51, 1991.

[Data 16] Data61, CSIRO. "Hyperspectral Data Sets". http://scyllarus.research.nicta.com.au/data/, June 2016. Retrieved 2016-10-09.

[De C 00] O. A. De Carvalho and P. R. Meneses. "Spectral correlation mapper (SCM): an improvement on the spectral angle mapper (SAM)". In: *Summaries, Ninth JPL Airborne Earth Science Workshop*, Jet Propulsion Lab, California Institute of Technology, Pasadena, CA, Feb. 2000.

[Demi 09] B. Demir, A. Celebi, and S. Erturk. "A low-complexity approach for the color display of hyperspectral remote-sensing images using one-bit-transform-based band selection". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 47, No. 1, pp. 97–105, 2009.

[Denn 04] P. Dennison, K. Halligan, and D. Roberts. "A comparison of error metrics and constraints for multiple endmember spectral mixture analysis and spectral angle mapper". *Remote Sensing of Environment*, Vol. 93, No. 3, pp. 359–367, 2004.

[Di Z 86] S. Di Zenzo. "A note on the gradient of a multi-image". *Computer vision, graphics, and image processing*, Vol. 33, No. 1, pp. 116–125, 1986.

[Dinh 11] C. V. Dinh, R. Leitner, P. Paclik, M. Loog, and R. P. Duin. "SEDMI: Saliency based edge detection in multispectral images". *Image and Vision Computing*, Vol. 29, No. 8, pp. 546–556, 2011.

[Dong 05] G. Dong and M. Xie. "Color clustering and learning for image segmentation based on neural networks". *IEEE Transactions on Neural Networks*, Vol. 16, No. 4, pp. 925–936, 2005.

[Drew 94] C. Drewniok. "Multi-spectral edge detection. Some experiments on data from Landsat-TM". *International Journal of Remote Sensing*, Vol. 15, No. 18, pp. 3743–3765, 1994.

[Du 04] Y. Du, C.-I. Chang, H. Ren, C.-C. Chang, J. O. Jensen, and F. M. D'Amico. "New hyperspectral discrimination measure for spectral characterization". *Optical Engineering*, Vol. 43, No. 8, pp. 1777–1786, Aug. 2004.

[Du 08] Q. Du and H. Yang. "Similarity-based unsupervised band selection for hyperspectral image analysis". *IEEE Geoscience and Remote Sensing Letters*, Vol. 5, No. 4, pp. 564–568, 2008.

[Dund 04] M. Dundar and D. Landgrebe. "Toward an Optimal Supervised Classifier for the Analysis of Hyperspectral Data". *IEEE Transaction on Geoscience and Remote Sensing*, Vol. 42, No. 1, pp. 271–277, 2004.

[Elli 06] G. Ellis and A. Dix. "Enabling automatic clutter reduction in parallel coordinate plots". *IEEE Transactions on Visualization and Computer Graphics*, Vol. 12, No. 5, pp. 717–724, 2006.

[Elma 12] G. Elmasry, D. F. Barbin, D.-W. Sun, and P. Allen. "Meat Quality Evaluation by Hyperspectral Imaging Technique: An Overview". *Critical Reviews in Food Science and Nutrition*, Vol. 52, No. 8, pp. 689–711, 2012.

[Este 96] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.* "A density-based algorithm for discovering clusters in large spatial databases with noise.". In: *International Conference on Knowledge Discovery and Data Mining*, pp. 226–231, American Association for Artificial Intelligence, Portland, OR, Aug. 1996.

[Esti 99] V. Estivill-Castro. "Sorting and Order Statistics". In: M. J. Atallah, Ed., *Algorithms and theory of computation handbook*, Chap. 3, CRC Press, 1999.

[Evan 06] A. N. Evans and X. U. Liu. "A morphological gradient approach to color edge detection". *IEEE Transactions on Image Processing*, Vol. 15, No. 6, pp. 1454–1463, 2006.

[Exel 16] Exelis Visual Information Solutions. "ENVI". http://www.harrisgeospatial.com/ProductsandSolutions/GeospatialProducts/ENVI.aspx, Feb. 2016. Retrieved 2016-10-09.

[Fauv 06] M. Fauvel, J. Chanussot, and J. A. Benediktsson. "Kernel principal component analysis for feature reduction in hyperspectrale images analysis". In: *Nordic Signal Processing Symposium*, pp. 238–241, IEEE, Reykjavik, June 2006.

[Fauv 07] M. Fauvel. *Spectral and Spatial Methods for the Classification of Urban Remote Sensing Data*. PhD thesis, Institut National Polytechnique de Grenoble - INPG ; Université d'Islande, Nov. 2007.

[Fauv 09] M. Fauvel, J. Chanussot, and J. A. Benediktsson. "Kernel principal component analysis for the classification of hyperspectral remote sensing data over urban areas". *EURASIP Journal on Advances in Signal Processing*, Vol. 2009, pp. 1–14, 2009.

[Fauv 13] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton. "Advances in Spectral-Spatial Classification of Hyperspectral Images". *Proceedings of the IEEE*, Vol. 101, No. 3, pp. 652–675, March 2013.

[Felz 04] P. F. Felzenszwalb and D. P. Huttenlocher. "Efficient graph-based image segmentation". *International Journal of Computer Vision*, Vol. 59, No. 2, pp. 167–181, 2004.

[Fish 98] J. Fisher, M. M. Baumback, J. H. Bowles, J. M. Grossmann, and J. A. Antoniades. "Comparison of low-cost hyperspectral sensors". In: *International Symposium on Optical Science, Engineering, and Instrumentation*, pp. 23–30, SPIE, San Diego, CA, July 1998.

[Fonv 13] J. Fonville, C. Carter, L. Pizarro, R. Steven, A. Palmer, R. Griffiths, P. Lalor, J. Lindon, J. Nicholson, E. Holmes, and J. Bunch. "Hyperspectral Visualization of Mass Spectrometry Imaging Data". *Analytical Chemistry*, Vol. 85, No. 3, pp. 1415–1423, 2013.

[Fost 06] D. H. Foster, K. Amano, S. Nascimento, and M. J. Foster. "Frequency of metamerism in natural scenes". *Journal of the Optical Society of America A*, Vol. 23, No. 10, pp. 2359–2372, 2006.

[Fost 16] D. H. Foster, K. Amano, and S. M. Nascimento. "Time-lapse ratios of cone excitations in natural scenes". *Vision Research*, Vol. 120, pp. 45–60, 2016.

[Free 10] D. Freedman and P. Kisilev. "KDE paring and a faster mean shift algorithm". *SIAM Journal on Imaging Sciences*, Vol. 3, No. 4, pp. 878–903, 2010.

[Frit 94] B. Fritzke. "Growing cell structures–a self-organizing network for unsupervised and supervised learning". *Neural Networks*, Vol. 7, No. 9, pp. 1441–1460, 1994.

[Fuch 09] R. Fuchs and H. Hauser. "Visualization of multi-variate scientific data". *Computer Graphics Forum*, Vol. 28, No. 6, pp. 1670–1690, 2009.

[Fuku 75] K. Fukunaga and L. Hostetler. "The estimation of the gradient of a density function, with applications in pattern recognition". *IEEE Transactions on Information Theory*, Vol. 21, No. 21, pp. 32–40, 1975.

[Gat 00] N. Gat. "Imaging spectroscopy using tunable filters: a review". In: *Wavelet Applications VII*, pp. 50–64, SPIE, Orlando, Fl, Apr. 2000.

[Geel 14] B. Geelen, N. Tack, and A. Lambrechts. "A compact snapshot multispectral imager with a monolithically integrated per-pixel filter mosaic". In: *Advanced Fabrication Technologies for Micro/Nano Optics and Photonics VII*, p. 89740L, SPIE, San Francisco, CA, Feb. 2014.

[Geor 03] B. Georgescu, I. Shimshoni, and P. Meer. "Mean shift based clustering in high dimensions: A texture classification example". In: *IEEE International Conference on Computer Vision*, pp. 456–463, IEEE, Nice, Oct. 2003.

[Geor 07] G. T. Georgiev and J. J. Butler. "Long-term calibration monitoring of Spectralon diffusers BRDF in the air-ultraviolet". *Applied Optics*, Vol. 46, No. 32, pp. 7892–7899, Nov. 2007.

[Geve 12] T. Gevers, A. Gijsenij, J. Van de Weijer, and J.-M. Geusebroek. *Color in Computer Vision: Fundamentals and Applications*. John Wiley & Sons, 2012.

[Gonz 08] R. Gonzalez, R. Woods, and S. Eddins. *Digital Image Processing*. Prentice Hall Press, 3 Ed., 2008.

[Gopp 95] J. Göppert and W. Rosenstiel. "Topological interpolation in SOM by affine transformations". In: *European Symposium on Artificial Neural Networks*, pp. 15–20, Brussels, Apr. 1995.

[Gorr 12] J. Gorricha and V. Lobo. "Improvements on the visualization of clusters in geo-referenced data using Self-Organizing Maps". *Computers & Geosciences*, Vol. 43, pp. 177–186, 2012.

[Gowe 07] A. Gowen, C. O'Donnell, P. Cullen, G. Downey, and J. Frias. "Hyperspectral imaging – an emerging process analytical tool for food quality and safety control". *Trends in Food Science & Technology*, Vol. 18, No. 12, pp. 590–598, 2007.

[Grad 06] L. Grady. "Random Walks for Image Segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, pp. 1768–1783, 2006.

[Gree 05] R. L. Greenman, S. Panasyuk, X. Wang, T. E. Lyons, T. Dinh, L. Longoria, J. M. Giurini, J. Freeman, L. Khaodhiar, and A. Veves. "Early changes in the skin microcirculation and muscle metabolism of the diabetic foot". *The Lancet*, Vol. 366, No. 9498, pp. 1711–1717, 2005.

[Gree 98] R. O. Green, M. L. Eastwood, C. M. Sarture, T. G. Chrien, M. Aronsson, B. J. Chippendale, J. A. Faust, B. E. Pavri, C. J. Chovit, M. Solis, M. R. Olah, and O. Williams. "Imaging Spectroscopy and the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS)". *Remote Sensing of Environment*, Vol. 65, No. 3, pp. 227–248, 1998.

[Guo 06] B. Guo, S. R. Gunn, R. Damper, and J. Nelson. "Band selection for hyperspectral image classification using mutual information". *IEEE Geoscience and Remote Sensing Letters*, Vol. 3, No. 4, pp. 522–526, 2006.

[Guti 10] A. Gutiérrez-Rodríguez, M. Medina-Pérez, J. Martínez-Trinidad, J. Carrasco-Ochoa, and M. García-Borroto. "New dissimilarity measures for ultraviolet spectra identification". In: *Mexican Conference on Pattern Recognition*, pp. 220–229, Springer, Puebla, Sep. 2010.

[Habi 15] N. Habili and J. Oorloff. "Scyllarus™: From Research to Commercial Software". In: *Australasian Software Engineering Conference*, pp. 119–122, IEEE, Adelaide, S.A., Sep. 2015.

[Habo 04] D. Haboudane, J. R. Miller, E. Pattey, P. J. Zarco-Tejada, and I. B. Strachan. "Hyperspectral vegetation indices and novel algorithms for predicting green LAI of crop canopies: Modeling and validation in the context of precision agriculture". *Remote sensing of environment*, Vol. 90, No. 3, pp. 337–352, 2004.

[Hedj 13] R. Hedjam and M. Cheriet. "Historical document image restoration using multispectral imaging system". *Pattern Recognition*, Vol. 46, No. 8, pp. 2297–2312, 2013.

[Hedj 15] R. Hedjam and M. Cheriet. "Multispectral Images of Historical Documents". http://www.synchromedia.ca/databases/MSI-HISTODOC, 2015. Retrieved 2016-08-14.

[Hegy 15] A. Hegyi and J. Martini. "Hyperspectral imaging with a liquid crystal polarization interferometer". *Optical Express*, Vol. 23, No. 22, pp. 28742–28754, Nov. 2015.

[Hein 12] J. Heinrich, Y. Luo, A. E. Kirkpatrick, H. Zhang, and D. Weiskopf. "Evaluation of a bundling technique for parallel coordinates". In: *International Conference on Information Visualization Theory and Applications*, pp. 594–602, Scitepress, Rome, Feb. 2012.

[Hein 13] J. Heinrich and D. Weiskopf. "State of the Art of Parallel Coordinates". In: *Eurographics 2013 - State of the Art Reports*, Eurographics Association, Girona, May 2013.

[Hend 63] S. Henderon and D. Hodgkiss. "The Spectral Energy Distribution of Daylight". *British Journal of Applied Physics*, Vol. 14, No. 3, pp. 125–131, March 1963.

[Hend 64] S. Henderon and D. Hodgkiss. "The Spectral Energy Distribution of Daylight". *British Journal of Applied Physics*, Vol. 15, No. 8, pp. 947–952, Aug. 1964.

[Herr 12] E. Herrala. "Imaging Spectrometer". Patent Application, May 2012. U.S. 13/275698.

[Holz 03] S. Holzwarth, M. Habermeyer, R. Richter, A. Hausold, S. Thiemann, and P. Strobl. "HySens-DAIS 7915/ROSIS imaging spectrometers at DLR". In: *EARSeL Workshop on Imaging Spectroscopy*, pp. 3–14, European Association of Remote Sensing Laboratories, Herrsching, May 2003.

[Homa 04] S. Homayouni and M. Roux. "Hyperspectral image analysis for material mapping using spectral matching". In: *ISPRS Congress*, pp. B7,49–54, International Society for Photogrammetry and Remote Sensing, Istanbul, July 2004.

[Huan 08] X. Huang and L. Zhang. "An adaptive mean-shift analysis approach for object extraction and classification from urban hyperspectral imagery". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 46, No. 12, pp. 4173–4185, 2008.

[Hube 81] P. Huber. *Robust Statistics*. Wiley, 1981.

[Hugh 13] J. Hughes, A. van Dam, M. McGuire, D. F. Sklar, J. Foley, S. Feiner, and K. Akeley. *Computer graphics: principles and practice*. Addison-Wesley, 3rd Ed., 2013.

[Huyn 10a] C. P. Huynh and A. Robles-Kelly. "Hyperspectral imaging for skin recognition and biometrics". In: *IEEE International Conference on Image Processing*, pp. 2325–2328, IEEE, Hong Kong, Sep. 2010.

[Huyn 10b] C. P. Huynh and A. Robles-Kelly. "A probabilistic approach to spectral unmixing". In: *Structural, Syntactic, and Statistical Pattern Recognition*, pp. 344–353, Springer, 2010.

[Ifar 00] A. Ifarraguerri and C.-I. Chang. "Unsupervised hyperspectral image analysis with projection pursuit". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 38, No. 6, pp. 2529–2538, 2000.

[Inse 90] A. Inselberg and B. Dimsdale. "Parallel coordinates: a tool for visualizing multi-dimensional geometry". In: *IEEE Conference on Visualization*, pp. 361–378, IEEE, San Francisco, CA, Oct. 1990.

[Jaco 05] N. Jacobson and M. Gupta. "Design goals and solutions for display of hyperspectral images". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 43, No. 11, pp. 2684–2692, 2005.

[Jaco 07] N. P. Jacobson, M. R. Gupta, and J. B. Cole. "Linear fusion of image sets for display". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 45, No. 10, pp. 3277–3288, 2007.

[Jia 99] X. Jia and J. A. Richards. "Segmented principal components transformation for efficient hyperspectral remote-sensing image display and classification". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 37, No. 1, pp. 538–542, 1999.

[Jian 04] Y. Jiang and Z.-H. Zhou. "SOM ensemble-based image segmentation". *Neural Processing Letters*, Vol. 20, No. 3, pp. 171–178, 2004.

[Jime 99] L. Jimenez and D. Landgrebe. “Hyperspectral Data Analysis and Feature Reduction via Projection Pursuit”. *IEEE Transaction on Geoscience and Remote Sensing*, Vol. 37, No. 6, pp. 2653–2667, 1999.

[John 07] W. R. Johnson, D. W. Wilson, W. Fink, M. Humayun, and G. Bearman. “Snapshot hyperspectral imaging in ophthalmology”. *Journal of Biomedical Optics*, Vol. 12, No. 1, pp. 014036–014036–7, 2007.

[Jord 08] J. Jordan, S. Helwig, and R. Wanka. “Social interaction in particle swarm optimization, the ranked FIPS, and adaptive multi-swarms”. In: *Genetic and Evolutionary Computation Conference*, pp. 49–56, ACM, Atlanta, GA, July 2008.

[Jord 10] J. Jordan and E. Angelopoulou. “Gerbil - A Novel Software Framework for Visualization and Analysis in the Multispectral Domain”. In: *Vision, Modeling and Visualization*, pp. 259–266, Eurographics Association, Siegen, Nov. 2010.

[Jord 11] J. Jordan and E. Angelopoulou. “Edge detection in multispectral images using the n-dimensional self-organizing map”. In: *IEEE International Conference on Image Processing*, pp. 3181–3184, IEEE, Brussels, Sep. 2011.

[Jord 12a] J. Jordan. “Supervised Multispectral Image Segmentation”. http://www5.cs.fau.de/research/data/msseg/, Sep. 2012. Retrieved 2016-10-09.

[Jord 12b] J. Jordan and E. Angelopoulou. “Supervised Multispectral Image Segmentation With Power Watersheds”. In: *IEEE International Conference on Image Processing*, pp. 1585–1589, IEEE, Orlando, FL, Sep. 2012.

[Jord 13a] J. Jordan and E. Angelopoulou. “Hyperspectral Image Visualization with a 3-D Self-organizing Map”. In: *Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*, pp. 1–4, IEEE, Gainesville, FL, June 2013.

[Jord 13b] J. Jordan and E. Angelopoulou. “Mean-shift Clustering for Interactive Multispectral Image Analysis”. In: *IEEE International Conference on Image Processing*, pp. 3790–3794, IEEE, Melbourne, Sep. 2013.

[Jord 14] J. Jordan, E. Angelopoulou, and A. Robles-Kelly. “An unsupervised material learning method for imaging spectroscopy”. In: *International Joint Conference on Neural Networks*, pp. 2428–2435, IEEE, Beijing, July 2014.

[Jord 16a] J. Jordan. “Gerbil Hyperspectral Visualization and Analysis Framework”. http://gerbilvis.org, Sep. 2016. Retrieved 2016-10-09.

[Jord 16b] J. Jordan, E. Angelopoulou, and A. Maier. “A Novel Framework for Interactive Visualization and Analysis of Hyperspectral Image Data”. *Journal of Electrical and Computer Engineering*, Vol. 2016, No. 2635124, 2016.

[Joye 03] W. A. Joye and E. Mandel. “New features of SAOImage DS9”. In: *Astronomical Data Analysis Software and Systems XII*, pp. 489–492, 2003.

[Kaku 07] P. Kakumanu, S. Makrogiannis, and N. Bourbakis. "A survey of skin-color modeling and detection methods". *Pattern Recognition*, Vol. 40, No. 3, pp. 1106–1122, 2007.

[Kesh 02] N. Keshava and J. F. Mustard. "Spectral unmixing". *IEEE Signal Processing Magazine*, Vol. 19, No. 1, pp. 44–57, 2002.

[Kesh 04] N. Keshava. "Distance metrics and band selection in hyperspectral processing with applications to material identification and spectral libraries". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 42, No. 7, pp. 1552–1565, 2004.

[Khan 12] R. Khan, A. Hanbury, J. Stöttinger, and A. Bais. "Color based skin classification". *Pattern Recognition Letters*, Vol. 33, No. 2, pp. 157–163, 2012.

[Khul 99] S. Khuller and B. Raghavachari. "Advanced Combinatorial Algorithms". In: M. J. Atallah, Ed., *Algorithms and theory of computation handbook*, Chap. 7, CRC Press, 1999.

[Kian 97] M. Y. Kiang, U. R. Kulkarni, M. Goul, A. Philippakis, R. Chi, and E. Turban. "Improving the effectiveness of self-organizing map networks using a circular Kohonen layer". In: *Hawaii International Conference on System Sciences*, pp. 521–529, IEEE, Wailea, HI, Jan. 1997.

[Kim 10] S. Kim, S. Zhuo, F. Deng, C. Fu, and M. Brown. "Interactive Visualization of Hyperspectral Images of Historical Documents". *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, No. 6, pp. 1441–1448, 2010.

[Kind 80] R. Kindermann and L. Snell. *Markov random fields and their applications*. *Contemporary Mathematics*, American Mathematical Society, 1980.

[Knut 92] D. E. Knuth. "Two notes on notation". *The American Mathematical Monthly*, Vol. 99, No. 5, pp. 403–422, 1992.

[Koho 01] T. Kohonen. *Self-organizing maps*. Vol. 30 of *Springer series in information sciences*, Springer, 3rd Ed., 2001.

[Koho 82] T. Kohonen. "Self-organized formation of topologically correct feature maps". *Biological Cybernetics*, Vol. 43, No. 1, pp. 59–69, 1982.

[Koho 96] T. Kohonen, J. Hynninen, J. Kangas, and J. Laaksonen. "SOM_PAK: The self-organizing map program package". Tech. Rep. A31, Helsinki University of Technology, Laboratory of Computer and Information Science, 1996.

[Kotw 10] K. Kotwal and S. Chaudhuri. "Visualization of hyperspectral images using bilateral filtering". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 48, No. 5, pp. 2308–2316, 2010.

[Kotw 13] K. Kotwal and S. Chaudhuri. "A novel approach to quantitative evaluation of hyperspectral image fusion techniques". *Information Fusion*, Vol. 14, No. 1, pp. 5–18, 2013.

[Koua 03] E. Koua. "Using self-organizing maps for information visualization and knowledge discovery in complex geospatial datasets". In: *International Cartographic Conference: Cartographic Renaissance*, pp. 1694–1702, International Cartographic Association, Durban, Aug. 2003.

[Krus 93] F. Kruse, A. Lefkoff, J. Boardman, K. Heidebrecht, A. Shapiro, P. Barloon, and A. Goetz. "The spectral image processing system (SIPS) - interactive visualization and analysis of imaging spectrometer data". *Remote Sensing of Environment*, Vol. 44, No. 2–3, pp. 145–163, 1993.

[Labi 13a] B. Labitzke, S. Bayraktar, and A. Kolb. "Generic visual analysis for multi- and hyperspectral image data". *Data Mining and Knowledge Discovery*, Vol. 27, No. 1, pp. 117–145, 2013.

[Labi 13b] B. Labitzke, M. Paltian, and A. Kolb. "Radviz-based visual analysis of multispectral images". In: *Colour and Visual Computing Symposium*, pp. 1–6, IEEE, Gjovik, Sep. 2013.

[Lee 06] S. Lee and R. G. Lathrop. "Subpixel analysis of Landsat ETM+ using self-organizing map (SOM) neural networks for urban land cover characterization". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 6, pp. 1642–1654, June 2006.

[Lee 07] J. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer, 2007.

[Lee 93] C. Lee and D. A. Landgrebe. "Analyzing high-dimensional multispectral data". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 31, No. 4, pp. 792–800, 1993.

[Lenn 01] M. Lennon, G. Mercier, M. C. Mouchot, and L. Hubert-Moy. "Independent component analysis as a tool for the dimensionality reduction and the representation of hyperspectral images". In: *IEEE International Geoscience and Remote Sensing Symposium*, pp. 2893–2895, IEEE, Sydney, July 2001.

[Li 08] H. Li, C.-W. Fu, and A. Hanson. "Visualizing Multiwavelength Astrophysical Data". *IEEE Transactions on Visualization and Computer Graphics*, Vol. 14, No. 6, pp. 1555–1562, 2008.

[Lian 12] H. Liang. "Advances in multispectral and hyperspectral imaging for archaeology and art conservation". *Applied Physics A*, Vol. 106, No. 2, pp. 309–323, 2012.

[Lill 14] T. Lillesand, R. W. Kiefer, and J. Chipman. *Remote sensing and image interpretation*. John Wiley & Sons, 7 Ed., 2014.

[Liu 06] Y. Liu, R. H. Weisberg, and C. N. Mooers. "Performance evaluation of the self-organizing map for feature extraction". *Journal of Geophysical Research: Oceans*, Vol. 111, No. C05018, 2006.

[Lloy 82] S. Lloyd. "Least squares quantization in PCM". *IEEE Transactions on Information Theory*, Vol. 28, No. 2, pp. 129–137, 1982.

[Lope 13] C. Lopez-Molina, B. De Baets, and H. Bustince. "Quantitative error measures for edge detection". *Pattern Recognition*, Vol. 46, No. 4, pp. 1125–1139, 2013.

[Lu 14] G. Lu and B. Fei. "Medical hyperspectral imaging: a review". *Journal of Biomedical Optics*, Vol. 19, No. 1, p. 010901, 2014.

[Mals 11] C. R. Malskies, E. Eibenberger, and E. Angelopoulou. "The Recognition of Ethnic Groups based on Histological Skin Properties". In: *Vision, Modeling, and Visualization*, pp. 353–360, Eurographics Association, Berlin, Oct. 2011.

[Mand 96] A. Manduca. "Multispectral image visualization with nonlinear projections". *IEEE Transactions on Image Processing*, Vol. 5, No. 10, pp. 1486–1490, Oct. 1996.

[Mano 08] D. Manolakis, R. Lockwood, and T. Cooley. "On the spectral correlation structure of hyperspectral imaging data". In: *IEEE International Geoscience and Remote Sensing Symposium*, pp. II–581, IEEE, Boston, MA, July 2008.

[Mao 96] J. Mao and A. K. Jain. "A self-organizing network for hyperellipsoidal clustering (HEC)". *IEEE Transactions on Neural Networks*, Vol. 7, No. 1, pp. 16–29, 1996.

[Marc 14] V. Marchiafava, G. Bartolozzi, C. Cucci, M. De Vita, and M. Picollo. "Colour measurements for monitoring the conservation of contemporary artworks". *Journal of the International Colour Association*, Vol. 13, pp. 36–42, 2014.

[Mart 14] M. A. Martínez, E. M. Valero, J. Hernández-Andrés, J. Romero, and G. Langfelder. "Combining transverse field detectors and color filter arrays to improve multispectral imaging systems". *Applied Optics*, Vol. 53, No. 13, pp. C14–C24, 2014.

[McLa 08] G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. *Wiley Series in Probability and Statistics*, Wiley-Interscience, 2008.

[Mign 12] M. Mignotte. "A bicriteria-optimization-approach-based dimensionality-reduction model for the color display of hyperspectral images". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 50, No. 2, pp. 501–513, 2012.

[Mitt 12] A. Mittal, S. Sofat, and E. Hancock. "Detection of edges in color images: A review and evaluative comparison of state-of-the-art techniques". In: *International Conference on Autonomous and Intelligent Systems*, pp. 250–259, Springer, Aviero, June 2012.

[Mori 08] T. Morimoto, T. Mihashi, and K. Ikeuchi. "Color restoration method based on spectral information using normalized cut". *International Journal of Automation and Computing*, Vol. 5, No. 3, pp. 226–233, 2008.

[Moun 11] G. Mountrakis, J. Im, and C. Ogole. "Support vector machines in remote sensing: A review". *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 66, No. 3, pp. 247–259, 2011.

[Nasc 16] S. M. Nascimento, K. Amano, and D. H. Foster. "Spatial distributions of local illumination color in natural scenes". *Vision Research*, Vol. 120, pp. 39–44, 2016.

[Nezh 11] E. Nezhadarya and R. K. Ward. "A new scheme for robust gradient vector estimation in color images". *IEEE Transactions on Image Processing*, Vol. 20, No. 8, pp. 2211–2220, 2011.

[Nico 07] B. M. Nicolai, K. Beullens, E. Bobelyn, A. Peirs, W. Saeys, K. I. Theron, and J. Lammertyn. "Nondestructive measurement of fruit and vegetable quality by means of NIR spectroscopy: A review". *Postharvest biology and technology*, Vol. 46, No. 2, pp. 99–118, 2007.

[Orti 13] A. Ortiz, J. Górriz, J. Ramírez, D. Salas-Gonzalez, and J. M. Llamas-Elvira. "Two fully-unsupervised methods for MR brain image segmentation using SOM-based strategies". *Applied Soft Computing*, Vol. 13, No. 5, pp. 2668–2682, 2013.

[Palm 14] G. Palmas, M. Bachynskyi, A. Oulasvirta, H. P. Seidel, and T. Weinkauf. "An edge-bundling layout for interactive parallel coordinates". In: *IEEE Pacific Visualization Symposium*, pp. 57–64, IEEE, Yokohama, March 2014.

[Pico 07] M. Picollo, M. Bacci, A. Casini, F. Lotti, M. Poggesi, and L. Stefani. "Hyperspectral image spectroscopy: a 2D approach to the investigation of polychrome surfaces". In: *Conservation Science*, pp. 10–11, Milan, May 2007.

[Polz 04] G. Pölzlbauer. "Survey and Comparison of Quality Measures for Self-Organizing Maps". In: *Workshop on Data Analysis*, pp. 67–82, Elfa Academic Press, Tatranská Polianka, June 2004.

[Purd 16] Purdue Research Foundation. "Hyperspectral Images". https://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html, 2016. Retrieved 2016-10-09.

[Rand 71] W. M. Rand. "Objective criteria for the evaluation of clustering methods". *Journal of the American Statistical Association*, Vol. 66, No. 336, pp. 846–850, 1971.

[Raub 02] A. Rauber, D. Merkl, and M. Dittenbach. "The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data". *IEEE Transactions on Neural Networks*, Vol. 13, No. 6, pp. 1331–1341, 2002.

[Ries 09] C. Riess, J. Jordan, and E. Angelopoulou. "A Common Framework for Ambient Illumination in the Dichromatic Reflectance Model". In: *IEEE International Conference on Computer Vision Workshops*, pp. 1939–1946, IEEE, Kyoto, Sep. 2009.

[Rive 93] J.-F. Rivest, P. Soille, and S. Beucher. "Morphological gradients". *Journal of Electronic Imaging*, Vol. 2, No. 4, pp. 326–336, 1993.

[Robi 05a] S. Robila. "Using spectral distances for speedup in hyperspectral image processing". *International Journal of Remote Sensing*, Vol. 26, No. 24, pp. 5629–5650, 2005.

[Robi 05b] S. Robila and A. Gershman. "Spectral matching accuracy in processing hyperspectral data". In: *International Symposium on Signals, Circuits and Systems*, pp. 163–166, IEEE, Iasi, July 2005.

[Robl 12] A. Robles-Kelly and C. P. Huynh. *Imaging spectroscopy for scene analysis*. Springer Science & Business Media, 2012.

[Robl 15] A. Robles-Kelly and J. Jordan. "Processing hyperspectral or multispectral image data". Feb. 2015. US Patent App. 15/118,643.

[Rose 98] K. Rose. "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems". *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2210–2239, 1998.

[Rubn 98] Y. Rubner, C. Tomasi, and L. J. Guibas. "A metric for distributions with applications to image databases". In: *International Conference on Computer Vision*, pp. 59–66, IEEE, Bombay, Jan. 1998.

[Ruzo 01] M. A. Ruzon and C. Tomasi. "Edge, junction, and corner detection using color distributions". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 11, pp. 1281–1295, 2001.

[Saga 94] H. Sagan. *Space-filling curves*. *Universitext*, Springer, New York, 1994.

[Scho 02] B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

[Shap 09] L. Shapira, S. Avidan, and A. Shamir. "Mode-detection via median-shift". In: *IEEE International Conference on Computer Vision*, pp. 1909–1916, IEEE, Kyoto, Sep. 2009.

[Shaw 02] G. Shaw and D. Manolakis. "Signal processing for hyperspectral image exploitation". *IEEE Signal Processing Magazine*, Vol. 19, No. 1, pp. 12–16, Jan. 2002.

[Shet 10] S. Shetty and N. Ahuja. "Supervised and unsupervised clustering with probabilistic shift". In: *European Conference on Computer Vision*, pp. 644–657, Springer, Heraklion, Sep. 2010.

[Shi 14] C. Shi and L. Wang. "Incorporating spatial information in spectral unmixing: A review". *Remote Sensing of Environment*, Vol. 149, pp. 70–87, 2014.

[Shi 97] J. Shi and J. Malik. "Normalized Cuts and Image Segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, pp. 888–905, 1997.

[Shir 08] S. Shirdhonkar and D. W. Jacobs. "Approximate earth mover's distance in linear time". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, Anchorage, AK, June 2008.

[Shre 13] D. Shreiner, G. Sellers, J. M. Kessenich, and B. M. Licea-Kane. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.3*. Addison-Wesley Professional, 8 Ed., 2013.

[Sjob 09] M. Sjöberg and J. Laaksonen. "Optimal Combination of SOM Search in Best-Matching Units and Map Neighborhood". In: *International Workshop on Advances in Self-Organizing Maps*, pp. 281–289, Springer, St. Augustine, FL, June 2009.

[Skup 13] A. Skupin, J. R. Biberstine, and K. Börner. "Visualizing the Topical Structure of the Medical Sciences: A Self-Organizing Map Approach". *PLoS ONE*, Vol. 8, No. 3, pp. 1–16, 2013.

[Stef 07] L. Stefani. "Non-invasive analysis on works of art". http://www.ifac.cnr.it/durer/, March 2007. Retrieved 2016-07-06.

[Stok 96] M. Stokes, M. Anderson, S. Chandrasekar, and R. Motta. "A Standard Default Color Space for the Internet-sRGB". Tech. Rep. http://www.color.org/sRGB.xalter, Microsoft and Hewlett-Packard, Nov. 1996. Retrieved 2016-10-09.

[Surk 11] M. Šurkala, K. Mozdřeň, R. Fusek, and E. Sojka. "Hierarchical Blurring Mean-Shift". In: *International Conference on Advanced Concepts for Intelligent Vision Systems*, pp. 228–238, Springer, Ghent, Aug. 2011.

[Surk 12] M. Šurkala, K. Mozdřeň, R. Fusek, and E. Sojka. "Hierarchical Evolving Mean-Shift". In: *IEEE International Conference on Image Processing*, pp. 1593–1596, IEEE, Orlando, FL, Sep. 2012.

[Taka 01] M. Takatsuka. "An application of the self-organizing map and interactive 3-D visualization to geospatial data". In: *Sixth International Conference on GeoComputation*, Sep. 2001.

[Talb 09] E.-G. Talbi. *Metaheuristics: From design to implementation*. John Wiley & Sons, 2009.

[Tama 99] R. Tamassia and B. Cantrill. "Basic data structures". In: M. J. Atallah, Ed., *Algorithms and theory of computation handbook*, Chap. 4, CRC Press, 1999.

[Tara 09] Y. Tarabalka, J. A. Benediktsson, and J. Chanussot. "Spectral–spatial classification of hyperspectral imagery based on partitional clustering techniques". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 47, No. 8, pp. 2973–2987, 2009.

[Tara 10] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson. "Segmentation and classification of hyperspectral images using watershed transformation". *Pattern Recognition*, Vol. 43, No. 7, pp. 2367–2379, 2010.

[Tasd 09] K. Tasdemir and E. Merényi. "Exploiting data topology in visualization and clustering of self-organizing maps". *IEEE Transactions on Neural Networks*, Vol. 20, No. 4, pp. 549–562, 2009.

[Tasd 11] K. Tasdemir, P. Milenov, and B. Tapsall. "Topology-based hierarchical clustering of self-organizing maps". *IEEE Transactions on Neural Networks*, Vol. 22, No. 3, pp. 474–485, 2011.

[Tene 00] J. B. Tenenbaum, V. de Silva, and J. C. Langford. "A Global Geometric Framework for Nonlinear Dimensionality Reduction". *Science*, Vol. 290, No. 5500, pp. 2319–2323, Dec. 2000.

[Thom 10] D. R. Thompson, L. Mandrake, M. S. Gilmore, and R. Castaño. "Superpixel endmember detection". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 48, No. 11, pp. 4023–4033, Nov. 2010.

[Toiv 03] P. Toivanen, J. Ansamäki, J. Parkkinen, and J. Mielikäinen. "Edge detection in multispectral images using the self-organizing map". *Pattern Recognition Letters*, Vol. 24, No. 16, pp. 2987–2994, 2003.

[Trah 93] P. Trahanias and A. Venetsanopoulos. "Color edge detection using vector order statistics". *IEEE Transactions on Image Processing*, Vol. 2, No. 2, pp. 259–264, 1993.

[Tsag 05] V. Tsagaris, V. Anastassopoulos, and G. A. Lampropoulos. "Fusion of hyperspectral data using segmented PCT for color representation and classification". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 43, No. 10, pp. 2365–2375, 2005.

[Tyo 03] J. Tyo, A. Konsolakis, D. Diersen, and R. Olsen. "Principal-Components-Based Display Strategy for Spectral Imagery". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 41, No. 3, pp. 708–718, 2003.

[Ults 03] A. Ultsch. "Maps for the visualization of high-dimensional data spaces". In: *Workshop on Self organizing Maps*, pp. 225–230, Kyushu, Sep. 2003.

[Unit 16] United States Army Geospatial Center. "Hypercube". http://www.erdc.usace.army.mil/Media/FactSheets/FactSheetArticleView/tabid/9254/Article/610433/hypercube.aspx, May 2016. Retrieved 2016-10-09.

[US G 15] U.S. Geological Survey. "What are the best spectral bands to use for my study?". http://landsat.usgs.gov/best_spectral_bands_to_use.php, Sep. 2015. Retrieved 2016-07-31.

[Usti 00] S. L. Ustin and A. Trabucco. "Using hyperspectral data to assess forest structure". *Journal of Forestry*, Vol. 98, No. 6, pp. 47–49, 2000.

[Veda 08] A. Vedaldi and S. Soatto. "Quick shift and kernel methods for mode seeking". In: *European Conference on Computer Vision*, pp. 705–718, Springer, Marseille, Oct. 2008.

[Vesa 00a] J. Vesanto and E. Alhoniemi. "Clustering of the self-organizing map". *IEEE Transactions on Neural Networks*, Vol. 11, No. 3, pp. 586–600, 2000.

[Vesa 00b] J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas. "SOM toolbox for Matlab 5". Tech. Rep. A57, Helsinki University of Technology, Finland, 2000.

[Vesa 99] J. Vesanto. "SOM-based data visualization methods". *Intelligent data analysis*, Vol. 3, No. 2, pp. 111–126, 1999.

[Vill 03] T. Villmann, E. Merényi, and B. Hammer. "Neural maps in remote sensing image analysis". *Neural Networks*, Vol. 16, No. 3, pp. 389–403, 2003.

[Wang 06a] H. Wang and E. Angelopoulou. "Sensor band selection for multispectral imaging via average normalized information". *Journal of Real-Time Image Processing*, Vol. 1, No. 2, pp. 109–121, 2006.

[Wang 06b] J. Wang and C.-I. Chang. "Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis". *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 44, No. 6, pp. 1586–1600, 2006.

[Warm 08] F. Warmerdam. "The Geospatial Data Abstraction Library". In: G. B. Hall, M. G. Leahy, S. Balram, and S. Dragicevic, Eds., *Open Source Approaches in Spatial Data Handling*, pp. 87–104, Springer, 2008.

[Wija 11] D. Wijayasekara, O. Linda, and M. Manic. "CAVE-SOM: Immersive visual data mining using 3D Self-Organizing Maps". In: *International Joint Conference on Neural Networks*, pp. 2471–2478, IEEE, San Jose, CA, Aug. 2011.

[Wu 04] S. Wu and T. W. Chow. "Clustering of the self-organizing map using a clustering validity index based on inter-cluster and intra-cluster density". *Pattern Recognition*, Vol. 37, No. 2, pp. 175–188, 2004.

[Wu 07] K.-L. Wu and M.-S. Yang. "Mean shift-based clustering". *Pattern Recognition*, Vol. 40, No. 11, pp. 3035–3052, 2007.

[Wysz 00] G. Wyszecki and W. S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley-Interscience, 2 Ed., July 2000.

[Xu 05] R. Xu and D. Wunsch. "Survey of clustering algorithms". *IEEE Transactions on Neural Networks*, Vol. 16, No. 3, pp. 645–678, 2005.

[Yang 11] H. Yang, Q. Du, H. Su, and Y. Sheng. "An Efficient Method for Supervised Hyperspectral Band Selection". *IEEE Geoscience and Remote Sensing Letters*, Vol. 8, No. 1, pp. 138–142, Jan. 2011.

[Yasu 10] F. Yasuma, T. Mitsunaga, D. Iso, and S. K. Nayar. "Generalized Assorted Pixel Camera: Post-Capture Control of Resolution, Dynamic Range and Spectrum". *IEEE Transactions on Image Processing*, Vol. 19, No. 9, pp. 2241–2253, Sep. 2010.

[Yuha 92] R. H. Yuhas, A. F. H. Goetz, and J. W. Boardman. "Discrimination among semi-arid landscape endmembres using the Spectral Angle Mapper (SAM) algorithm". In: *Summaries, Third JPL Airborne Geoscience Workshop*, pp. 147–149, Jet Propulsion Lab, California Institute of Technology, Pasadena, CA, June 1992.

[Zhan 04] H. Zhang, J. E. Fritts, and S. A. Goldman. "An entropy-based objective evaluation method for image segmentation". In: *Storage and Retrieval Methods and Applications for Multimedia*, pp. 38–49, SPIE, San Jose, CA, Jan. 2004.

[Zhan 12] X. Zhang, P. Xiao, and X. Feng. "An Unsupervised Evaluation Method for Remotely Sensed Imagery Segmentation". *IEEE Geoscience and Remote Sensing Letters*, Vol. 9, No. 2, pp. 156–160, March 2012.

[Zhou 08] H. Zhou, X. Yuan, H. Qu, W. Cui, and B. Chen. "Visual clustering in parallel coordinates". *Computer Graphics Forum*, Vol. 27, No. 3, pp. 1047–1054, 2008.

[Zhu 07] Y. Zhu, P. K. Varshney, and H. Chen. "Evaluation of ICA based fusion of hyperspectral images for color display". In: *International Conference on Information Fusion*, pp. 1–7, IEEE, Quebec, QC, July 2007.