FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Automatische Zusammenfassung von Bild-Datensätzen

Automatische Gruppierung und Visualisierung nach der geographischen Lage

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Patrick Mayr
Matrikelnummer 1226745

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.tech. Eduard Gröller
Mitwirkung: Dipl.-Ing. Dr.tech. Johanna Schmidt

Wien, 1. April 2017

Patrick Mayr | Eduard Gröller

Automatic Summarization of Image Datasets

Automatic Clustering and Visualization according to geographical Locations

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Media Informatics and Visual Computing

by

Patrick Mayr
Registration Number 1226745

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.tech. Eduard Gröller
Assistance: Dipl.-Ing. Dr.tech. Johanna Schmidt

Vienna, 1st April, 2017

Patrick Mayr   Eduard Gröller

Erklärung zur Verfassung der Arbeit

Patrick Mayr
Favoritenstraße 9-11
1040 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. April 2017

Patrick Mayr

Danksagung

An dieser Stelle möchte ich mich besonders bei meiner Betreuerin Johanna Schmidt dafür bedanken, dass ich immer konstruktives Feedback und Denkanstöße erhalten habe, wenn es nötig war. Gleichzeitig wurde mir großer Freiraum gewährt, so dass ich schlussendlich sogar mein eigenes Thema im Rahmen dieser Bachelorarbeit umsetzen konnte.

Weiters möchte ich mich bei meinen Studienkollegen Bernhard Reischl und Ulrik Schremser für die Zurverfügungstellung ihrer Interrail Fotos bedanken.

Darüber hinaus möchte ich mich bei meiner Familie bedanken, die mich während meines Studiums unterstützt hat.

Acknowledgements

At this point I would like to thank my supervisor Johanna Schmidt that I always got constructive feedback and food for thought, if it was needed. At the same time big clearance was granted to me, so that I could finally even move my own subject within the scope of this bachelor thesis.

I would also like to thank my fellow student Bernhard Reischl and Ulrik Schremser for providing their Interrail photos.

In addition, I would like to thank my family for supporting me during my studies.

Kurzfassung

Seit der Erfindung der Digitalkamera nimmt die Anzahl der aufgenommenen Bildern immer noch zu. Besonders in den letzten Jahren, als die Mobiltelefonhersteller begannen ihren Mobiltelefonen Digitalkameras hinzuzufügen, endete das in einer Flut von Bildern in der heutigen Zeit. Das Ziel dieser Bachelorarbeit war es, einen Algorithmus zu implementieren, der automatisch einen gegebenen Datensatz von Bildern mit ihren geographischen Koordinaten zusammenfassen und eine repräsentative Bildercollage erstellt. Der Algorithmus hält sich dabei strikt an die Visualisierungspipeline und startet mit der Analyse des Bild-Datensatzes, gefolgt vom Filtern. Im nächsten Schritt werden die gefilterten Daten abgebildet. Hierzu wird eine symmetrische Version der Voronoi-Tessellation verwendet. Schlussendlich werden die Daten gerendert, wobei hier ein optionales Dissolve-Blending möglich ist. Wir testen die Implementierung mit zwei unterschiedlichen Bilddatensätzen, welche 35 beziehungsweise 154 Bilder beinhalten. Die Ergebnisse deuten darauf hin, dass die Implementierung abwechslungsreiche und ansprechende Bildercollagen für die getesteten Bilddatensätze erzeugt, aber dass es auch einige Extremfälle wie etwa irreführende Symmetrie gibt, die durch die Präattentive Wahrnehmung des visuellen Systems des Menschens ausgelöst wird.

Abstract

Since the innovation of the digital camera the number of pictures taken is still increasing. Especially in recent years, when mobile phone manufacturer began to attach digital cameras to their mobile phones, this ended up in a flood of pictures nowadays. The aim of this thesis is, to implement an algorithm that automatically summarizes a given dataset of geotagged images and creates a representative image collage. It is strictly based on the visualization pipeline and starts with the data analysis and filtering of the image dataset. Next, the filtered data are mapped using a symmetric version of the Voronoi tessellation, and are finally rendered using optional dissolve blending. We test the implementation on two different image datasets, containing 35 and 154 images, respectively. The results indicate that the implementation generates diverse and appealing image collages for the tested image datasets, but that there are also some extreme cases for example with misleading symmetry caused by the pre-attentive processing of the human visual system.

Contents

CHAPTER 1

Introduction

This bachelor thesis deals with the automatic summarization of image datasets, which contain the geographical location as additional information. The resulting summary based on the geographical location is visualized through an image collage based on *Voronoi* diagrams. The following sections give an overview on the motivation and goals of this topic, the used methodological approach, and the structure of the thesis.

1.1 Motivation

At the present time one interesting challenge is the rapidly growing number of data, ending up in so-called *Big Data* [CCS12, MCB^{+}11]. *Big Data* is problematic, because people struggle more and more with the task of getting useful information out of these huge amount of data. Especially in recent years, with the beginning of the digital age arround 2002 as described by Hilbert et al. [HL11], this increase of data ended up in a flood of pictures and videos nowadays, because mobile phone manufacturer began to attach digital cameras to their mobile phones and people also began to prefer digital over analog storage. So there is a need for methods that are capable to filter the relevant information out of this large amount of data. Pietzsch et al. [PSPT14], for example, deal with the interactive visualization and image processing of terabytes of image volumes. In this particular case the thesis is dealing with image data.

One of many use cases is, for example, an image search, which returns a huge set of images [SEZ13]. Such a large dataset is really hard to handle and to extract any useful information from. Therefore, people began to use basic data mining and pattern recognition techniques, such as clustering and categorization, to filter and classify the images, to get some information, respectively, and to split the datasets into smaller chunks. In recent years science refined methods based on these basic techniques, and even developed new scientific methods to handle this problem.

The most common goals of such summarization methods are the following:

1. Good overview of the dataset
2. Efficient implementation
3. Selection of a subset of representative images
4. Maximize information with respect to salient regions in a given area
5. Minimize blank areas, i.e., wasted space

Depending on the field of application, a good overview of the dataset can have a lot of different meanings. Therefore, basic techniques like clustering or tessellation were used in the first place and new advanced techniques like *Image Collages*, *Image Packing*, *Image Summarization*, or the *Illustration of Image Search Results* (see Section 2.2 to 2.5) were invented later on for different purposes.

Since image datasets can easily have the size of several gigabytes, an efficient implementation is necessary. In the *Illustration of Image Search Results* (see Section 2.5) for example it is really important, because the user wants to search and navigate through large image datasets in real-time.

The selection of representative images is a difficult task, because there are many different ways to get a representative description of an images dataset. For example, there are techniques that focus on the visual characteristics only, while others focus on the semantic information and others again try a hybrid approach by combining them. This selection task itself is also the key task of the classic *Image Summarization* (see Section 2.4) rather than the actual visualization of these images.

Finally, the resulting visualization should maximize the information in a given area and therefore minimize wasted space like blank areas. This is achieved by determining the salient regions for each of the representative images and arrange them in a way, that the salient regions do not or have a minimal overlap, while trying to maximize the coverage of the given area. Therefore, different saliency detection methods were invented, whereas some of them require prior knowledge of the image dataset, while others not. The latter, for example try to simulate the *pre-attentive* processing of the human visual system to determine the salient regions without prior knowledge.

Like the other *Image Collages* (see Section 2.2), the main focus of this thesis are especially on the goals 1, 4 and 5.

1.2 Problem Statement and Aim of the Thesis

As a starting point a set of images is provided. Each image contains the geographical location attached to its Exchangeable image file format (Exif) data.

The aim of this thesis is to implement an algorithm similar to the *ImageHive* [TSLX11] approach, based on the geographical location of the images. A subset of representative images from each cluster should be displayed in the collage, and it is allowed to cover unimportant information in the background of the images by other images. The focus of the project is on the useful arrangement of the images, and the optimal presentation of the relevant information, rather than on artistic merits.

1.3 Methodological Approach

The methodological approach is strictly based on the visualization pipeline. After loading the raw data through an graphical user interface, the raw data is analyzed and prepared for the filtering step. In the filtering step the data are filtered using *k-means* (see Section 2.1.1) as clustering method. In the next step the filtered data is mapped into a *Voronoi* diagram (see Section 2.1.2). Afterwards the mapped data is rendered with optional blending. As defined in the visualization pipeline, the user is able to interact with the application in each of these steps.

1.4 Structure of the Thesis

This chapter gave an introduction to the topic of the thesis, the underlying motivation, and the methodological approach. Chapter 2 reviews the state of the art in this field of information visualization and compares the different approaches. Chapter 3 explains the used methods and concepts in detail, which were necessary to generate the final visualization. Chapter 4 shows and compares resulting visualizations with different parameter settings. Chapter 5 reflects the thesis and gives a prospect on possible further developments.

CHAPTER 2

State of the Art

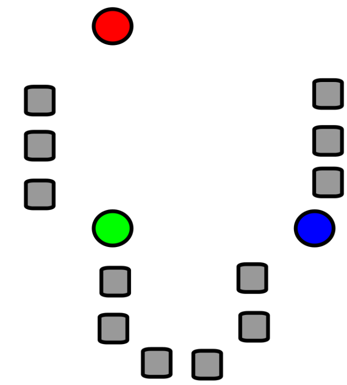
In this chapter the current state of the art concerning the main topics of this thesis are discussed. The first section (2.1) introduces some basic techniques like clustering and tessellation, which most of the advanced techniques later on use one way or another. Section 2.2 is dedicated to different approaches of creating an image collage and is the most important section for this bachelor thesis. Section 2.3 concentrates on the packing and layouting of an image dataset in a pre-defined space by using certain constraints, whereas section 2.4 describes methods that have the aim to extract a compact representative subset rather than visualize it. The last section (2.5) deals with the illustration of image search results.

2.1 Basic Techniques

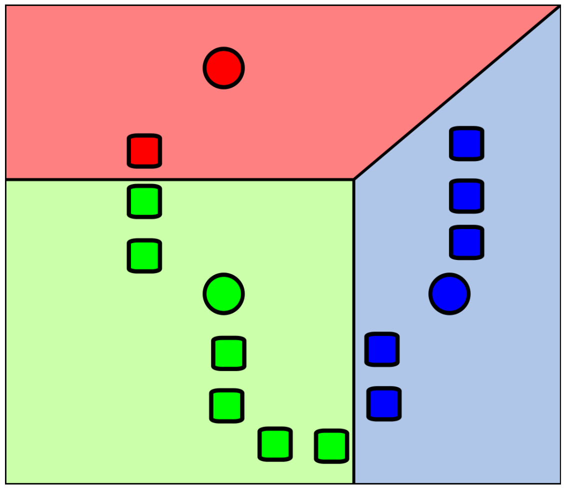
This section concentrates on basic techniques which were for example inherited from data mining and pattern recognition methods. These techniques were used in the early days and refined in recent years. Furthermore, they are nowadays the basis or part of most of the presented advanced techniques in the remaining sections. Clustering on the one hand allows to classify the images of an image dataset and split it into smaller chunks. Tessellation on the other hand is for example a way to create a layout to arrange the previously clustered image dataset.

2.1.1 Clustering

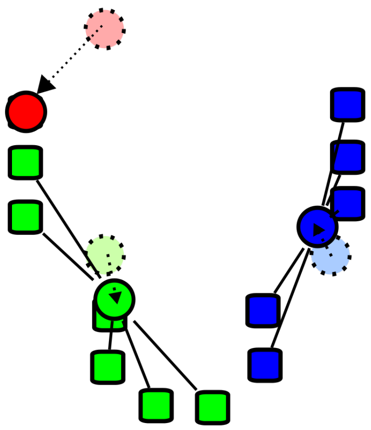
Clustering or cluster analysis is the task of partitioning a set of objects into so-called clusters, whereas objects within the same cluster are more similar to each other than to objects in other clusters. Clustering has a wide range of different techniques to analyze the clusters, which can be roughly divided into hierarchical and non-hierarchical approaches. Hierarchical approaches on the one hand construct a clustering tree and non-hierarchical approaches on the other hand assign each object in an iterative way to a cluster. For each of these approaches, the selection of the correct distance measure, which will determine the similarity of objects, and if necessary, additional parameters are crucial for a convincing result. The well-known *k-means* clustering algorithm [M+67] is an example for the non-hierarchical approaches and is described as next.



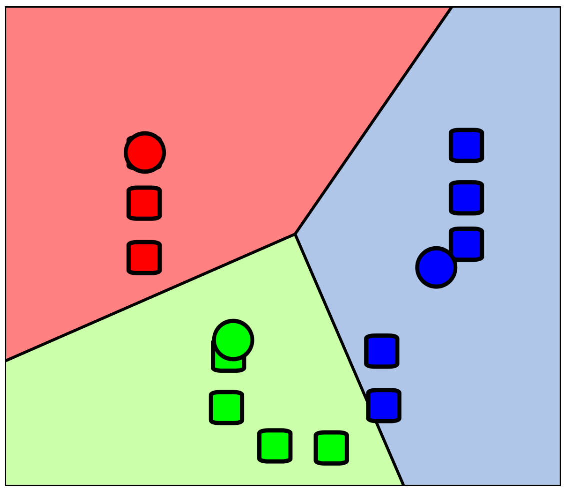
(a) k randomly generated initial means.



(b) k clusters are created by assining every observation to its nearest mean.



(c) The new means of the altered clusters are calculated.



(d) Step b and c are repeated until convergence has been reached.

Figure 2.1: Demonstration of the standard *k-means* clustering algorithm [Wikb].

The basic idea of the *k-means* clustering algorithm was originally adopted by MacQueen [M+67] and the standard algorithm was first proposed by Lloyd [Llo82]. This idea assumes that the number of clusters k is already known before starting the actual cluster analysis. It starts by choosing k different random means as initial cluster starting points. Afterwards, the *k-means* algorithm consists of a loop of an *Assignment* and an *Update* step. In the *Assignment* step each observation is assigned to the cluster with its least within-cluster sum of squares (WCSS). For the Euclidean distance, which is the standard distance measure, that means to assign the observations to its nearest mean respectively cluster centroid. Next, for each of the clusters, the new means respectively cluster centroids are calculated. The algorithm converges to a local optimum, when the assignments no longer change. Figure 2.1 demonstrates how the standard algorithm works.

In summary, the standard k-means algorithm consists of the following steps:

1. Initialization: Choose k random means from the dataset as cluster starting points.
2. Assignment step: Assign each observation to the cluster with its least WCSS (i.e. its nearest mean).
3. Update step: Calculate the new means of the altered clusters.

Due to its simplicity, the *k-means* algorithm has its limitations and many variations exist. Especially the accuracy, but also the speed of the *k-means* can be improved considerable by using different techniques to choose the initial cluster starting points. Arthur and Vassilvitskii [AV07] for example proposed their approximation algorithm *k-means++* in 2007. This algorithm can be used for choosing the initial values for the *k-means* clustering algorithm, which improves the final error of the *k-means* algorithm significant.

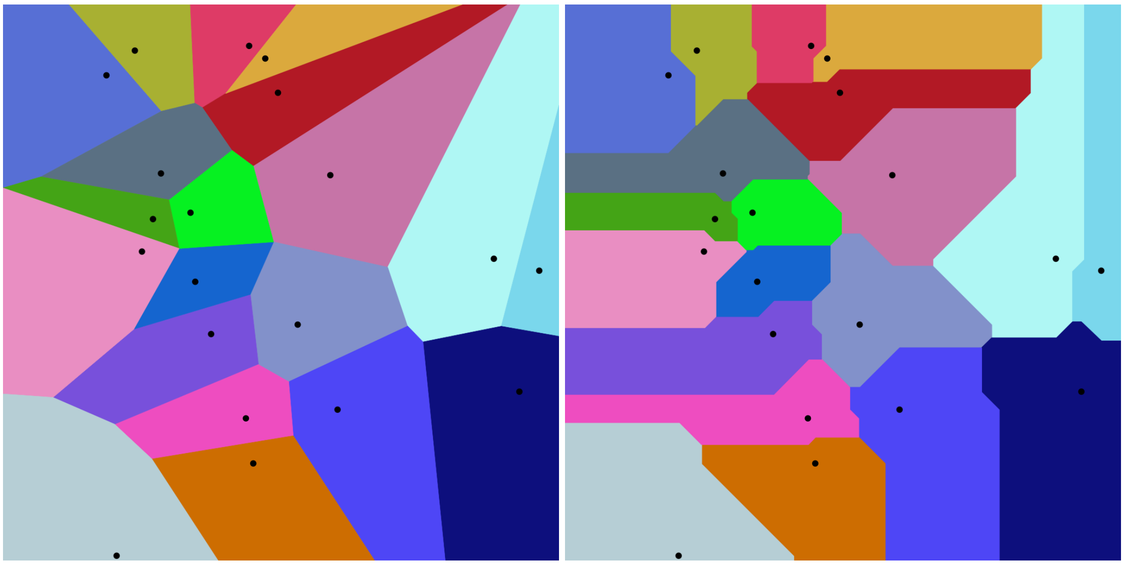
2.1.2 Tessellation

In this context tessellation or tiling is the task of arranging so-called *tiles* of varying shape to fill a 2-dimensional Euclidean plane without any gaps according to a pre-defined set of rules, which can vary depending on the actual use case. Common rules are that there are no gaps allowed between adjacent tiles, and also that it is not allowed that a corner of one tile lie along the edge of another [CBGS16]. Mathematically it is also possible to extend tessellations to other spaces than the Euclidean plane as Gullberg described [Gul97].

One Type of tessellation for example is the regular tessellation, which has both identical (congruent) regular tiles and identical vertices, which have the same angle between adjacent edges for every tile [Cox73]. There are only three shapes (the equilateral triangle, square, and regular hexagon) that can fulfill these conditions and can be duplicated infinitely to fill a plane with no gaps [Gul97].

A more important type of tessellation for this thesis is the so-called *Voronoi* tessellation or diagram named after Georgy Voronoi, who defined and studied the general n-dimensional case in 1908 [Vor08]. This approach was rediscovered over the last 200 years many times by different researcher and therefore it is also known as *Dirichlet* tessellation [Dir50] or *Thiessen* polygons [Thi11], depending on the field of application. Furthermore, it has also a close relation (i.e., the *Delauny* triangulation is the dual graph of the *Voronoi* tessellation) with the so-called *Delauny* triangulation named after Boris Delauny [Del34].

The simplest case of the *Voronoi* tessellation creates a tiling, where each tile respectively cell is defined as the set of points closest to one of the pre-defined seed points. Since each of these cells is obtained from the intersection of half-spaces, the resulting cell has always a convex shape. Note that different *Voronoi* diagrams are generated for different distance measures (see Figure 2.2).



(a) The Euclidean distance.

(b) The Manhattan distance.

Figure 2.2: Comparison of two different distance measures for the same set of seed points [Wikc].

There are also special types of the *Voronoi* tessellation like the centroidal Voronoi tessellation (CVT), which produces a tessellation whereas the seed point of each *Voronoi* cell is also its mean respectively center of mass. Lloyd's algorithm [Llo82] for example can be used to generate a CVT and its result can be considered as an optimal partition corresponding to an optimal distribution of generators. This algorithm works similar to the closely related *k-means* clustering algorithm (see Section 2.1.1) and only differs in its input, which is a continuous geometric region rather than a discrete set of points like for *k-means*. Therefore, it also uses the *Voronoi* tessellation in the *Assignment* step. Figure 2.3 shows three different solutions of the CVT for five points in a square. Furthermore, Du et al. [DFG99] give some applications of the CVT in their work.

2.2 Image Collages

This section deals with a more artistic way to summarize image datasets. In a so-called image collage one tries to arrange an image dataset according to pre-defined rules (e.g.

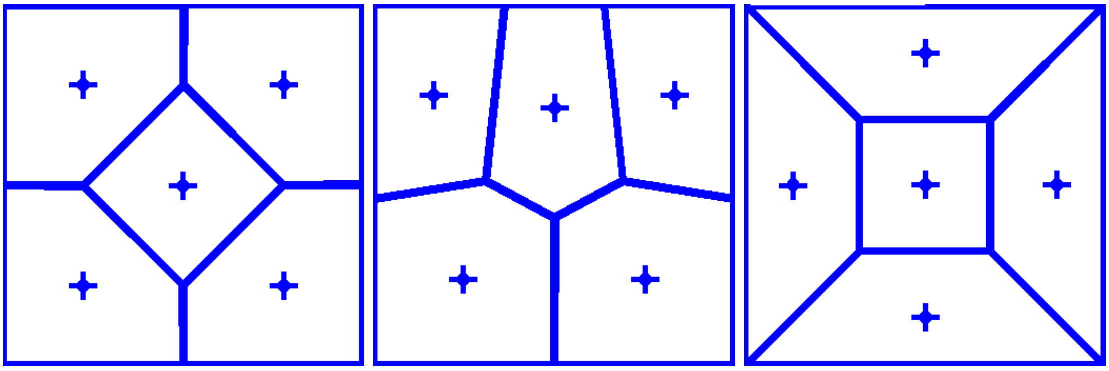


Figure 2.3: Three results of the CVT for five points in a square [Wika].

minimal blank space and overlap). For this thesis, the most relevant research and basis is the *ImageHive* approach, therefore it will be explained in detail in Section 2.2.1. Some of these approaches are even similar to real image collages created lifelike, so they look like some photographs that are simply glued on a canvas or were addtionally cut into shape with a scissors before.

2.2.1 ImageHive

ImageHive [TSLX11] is an approach which is able to interactively generate a summary of an existing image dataset based on a similarity measure (see Figure 2.4). This process is split into four steps.

Data Transformation

The first step starts with a clustering of the image dataset using a clustering algorithm such as k-means on features like color, or the edge histogram. This splits the image dataset into k different categories. For each of these categories, *ImageHive* chooses a number of representative images (e.g., images near the cluster center with some diversity among themselves). Afterwards, the salient regions for each representative image are determined using either a circle, or a rectangle as constraint.

Global Placement

In this step *ImageHive* constructs a graph layout for each cluster, where images are represented as nodes and the similarity correlation between them as edges. Later on the resulting graph lays out the images, which establishes the global placement (see Figure 2.5a). In this process the previously determined salient regions are used as a constraint, so that the images do not overlap.

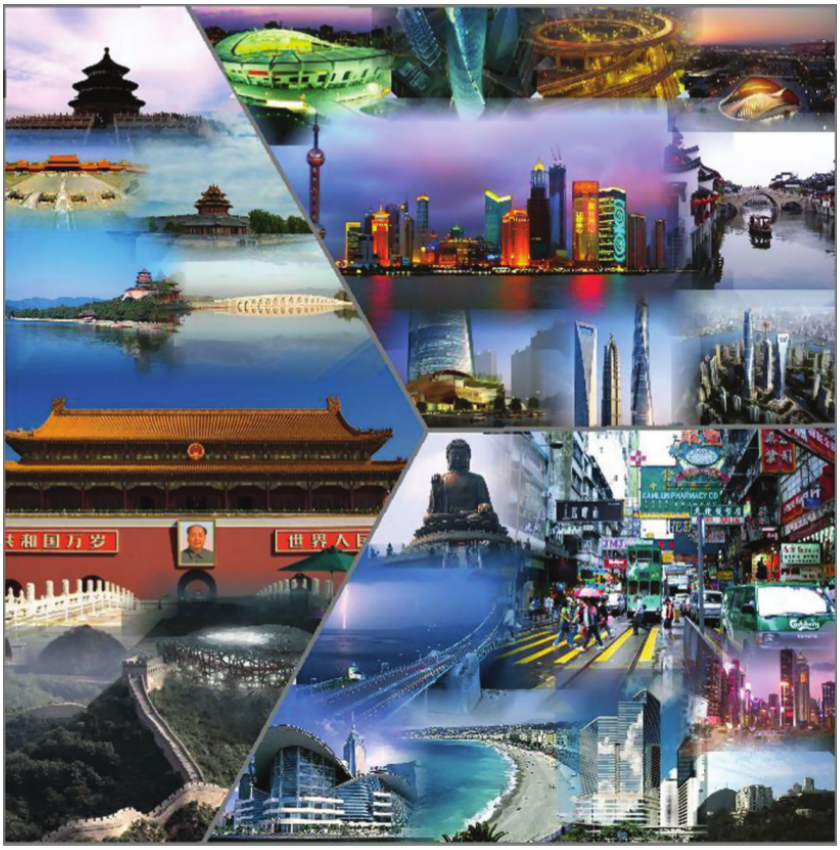


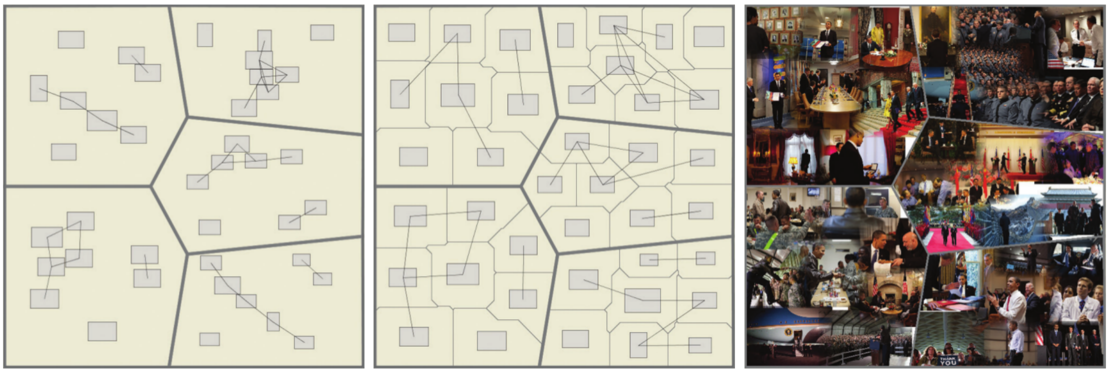
Figure 2.4: An image summary of Beijing (left), Shanghai (top right), and Hong Kong (bottom right) [TSLX11].

Local Adjustment

After the global placement the layout is locally refined by the so-called *Online Voronoi Tessellation*. Through this step the images are evenly distributed within a cluster while keeping the relationship from the graph layout model, which results in a maximized usage of the available layout area (see Figure 2.5b). The resulting layout distribute the images evenly while keeping their relationships (see Figure 2.5c).

Interactive Exploration

In a final step *ImageHive* allows the user to edit the automatic summarization in real-time and create new results by changing parameters such as image size, or the global placement of an image or cluster center. Finally, it is possible to blend the images, so that the resulting summarization has smooth transitions.



(a) Step 1 - global placement. (b) Step 2 - local adjustment. (c) The resulting layout.

Figure 2.5: The two-step layout of *ImageHive* [TSLX11].

2.2.2 AutoCollage

AutoCollage [RBHB06] is an approach that creates a collage of representative images from an image dataset (see Figure 2.6). It defines an automatic procedure which summarizes the main themes of the dataset. This procedure is based on the so-called *labelling problem*, which means that each pixel-location of the collage has to be assigned to a label. The goal of this algorithm is to find the best labelling. For that reason the authors introduce the so-called *collage energy*, which has to be minimized to get the best labelling. This energy consists of four terms such as *Representative Image Set Cost*, *Importance Cost*, *Transition Cost* and *Object Sensitivity Cost* to determine important regions in an image. Finally, they use a method called *α-Poisson Image Blending* to get a seamless transition between adjacent images in their collage.

2.2.3 Picture Collage

Similar to *AutoCollage*, *Picture Collage* [WQS$^+$06] deals with the problem of creating a collage from a group of images (see Figure 2.7). Within this summarization *Picture Collage* allows unimportant regions of the images to overlap to get a maximum of visible visual information. Furthermore, *Picture Collage* shows the most relevant regions of all images in the summary without any down-sampling or cropping. To achieve this the authors formulated the following four goals:

1. Maximize salience
2. Minimize blank space
3. Similar salience ratio balance
4. Orientation diversity

Figure 2.6: A collage of representative images created automatically by the *AutoCollage* approach [RBHB06].

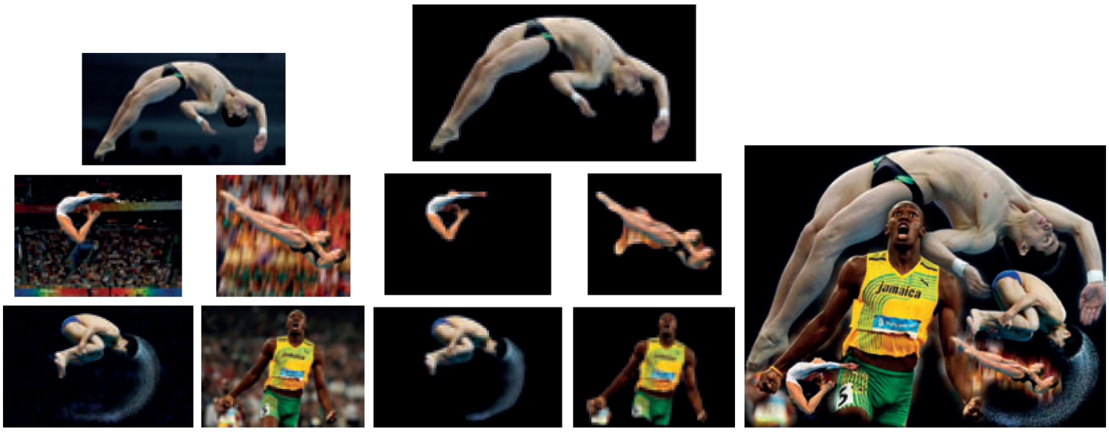
That means that the ideal collage should show as many salient regions as possible, while keeping approximately the same ratio of salience among the whole dataset. Furthermore, the blank space between the images has to be minimized to maximize the use of the canvas. Finally, introducing orientation diversity is necessary to imitate the style of a human created collage.

Since the placement order is similar to the related rectangle packing problem, which is NP-complete, the authors designed a sampling algorithm called Markov chain Monte Carlo (MCMC) to optimize this process and find a decent placement efficiently.

2.2.4 Puzzle-like Collage

Goferman et al. [GTZM10] propose a new method for the automatic construction of an image collage, which the authors call *Puzzle-like Collage* (see Figure 2.8). Unlike other approaches, which use rectangular regions of interest, this approach is based on the composition of regions of interest of varying shape in a puzzle-like manner. Therefore, the authors introduce an algorithm that is capable of extracting non-rectangular regions that coincide with meaningful information. Furthermore, they propose an algorithm, that keeps the balance between compactness and informativeness, while composing the region of interests to the final collage.

Figure 2.7: A result image created by the *Picture Collage* approach [WQS+06].



(a) Input images. (b) Regions of interest. (c) The resulting collage.

Figure 2.8: *Puzzle-like Collage* applied to an image collection of the 2008 Olympic games [GTZM10].

2.3 Image Packing

This section concentrates on the packing and layouting of an image dataset in a pre-defined space by using certain constraints. Such constraints are for example the minimization of the distance of similar images respectively the maximization of the correlation between dissimilarities among the data and their placement distance in the resulting layout. Furthermore, there are interaction techniques that allows the user to create a layout for some graphical primitives in an iterative and interactive way, without the need of sliders or similar interfaces.

2.3.1 Kernelized Sorting

Kernelized Sorting [QSS09] is an approach based on object matching, which is able to generalize the sorting of objects (see Figure 2.9). Object matching typically requires the definition of a similarity measure *between* classes, however the authors of this paper developed an approach which only needs a similarity measure *within* each class. This is achieved by using the so-called *Hilbert Schmidt Independence Criterion*, which maximizes the dependency between matched pairs. This way it is possible to perform matching without the need of a cross-domain similarity measure.

Figure 2.9: Layout of 284 images into a 'NIPS 2008' letter grid using kernelized sorting [QSS09].

2.3.2 Interactive By-example Design of Artistic Packing Layouts

Reinert et al. [RRS13] present an approach to pack 2D graphical primitives into a spatial layout without the need of sliders or similar interfaces, but instead by interactive placement using a subset of example primitives. To achieve a equal distribution of the primitives, the authors propose a novel generalization of the CVT, which should ensure an equal spacing between adjacent primitives.

Basically, their systems consists of an infinite loop of a forward layout step followed by an inverse layout step to achieve an artistic layout as result. The forward layout step takes care of the placement according to some rules and the inverse layout step refines the rules of the forward layout step and therefore the placement, according to the user interaction. A typical use case starts with a generic layout which maps primitives with similar features

Figure 2.10: Starting from a generic layout, the user moves three primitives (push-pins) to new locations. The first iteration leads to a layout sorted vertically by size and the second iteration sorts the layout additionally horizontal by brightness [RRS13].

to similar locations. Next, the user interactively moves a subset of example primitives (denoted by push-pin icon) to new locations, which are used afterwards in the forward layout step as new constraints (see Figure 2.10).

2.3.3 Organizing Visual Data in Structured Layout by Maximizing Similarity-Proximity Correlation

The work of Strong et al. [SJGE13] deals with the organization of visual data in structured layouts by maximizing the correlation between dissimilarities among the data and their placement distance in the resulting layout. In other words the problem can be described as maximizing the proximity-similarity correlation of the data. To achieve this, the authors propose an efficient greedy-based coarse-to-fine algorithm to compute an approximate solution for the specified problem. The result of this mapping is called the Max Correlation Map (MCM) (see Figure 2.11).

2.4 Image Summarization

This section concerns about the summarization of image datasets, which aims to get a compact representative subset rather than the actual visualization. Therefore, such techniques can be interesting to combine with other field of applications like the *Image Collages* or *Illustration of Image Search Results* (see Section 2.2 and 2.5) to improve already existing results. However, such methods for example try to find representative images from an image dataset by using the metadata of an image (e.g., tags), image similarity or a combination of both. Another field of research in this topic is, to determine the salient regions of a specific image and extract representative objects of it.

2.4.1 Context saliency based image summarization

Shi et al. [SWX^{+}09] propose a novel method for image summarization by using *context saliency* in cooperation with statistical saliency and geometric information, instead of visual saliency as importance measurement (see Figure 2.12). This approach is working

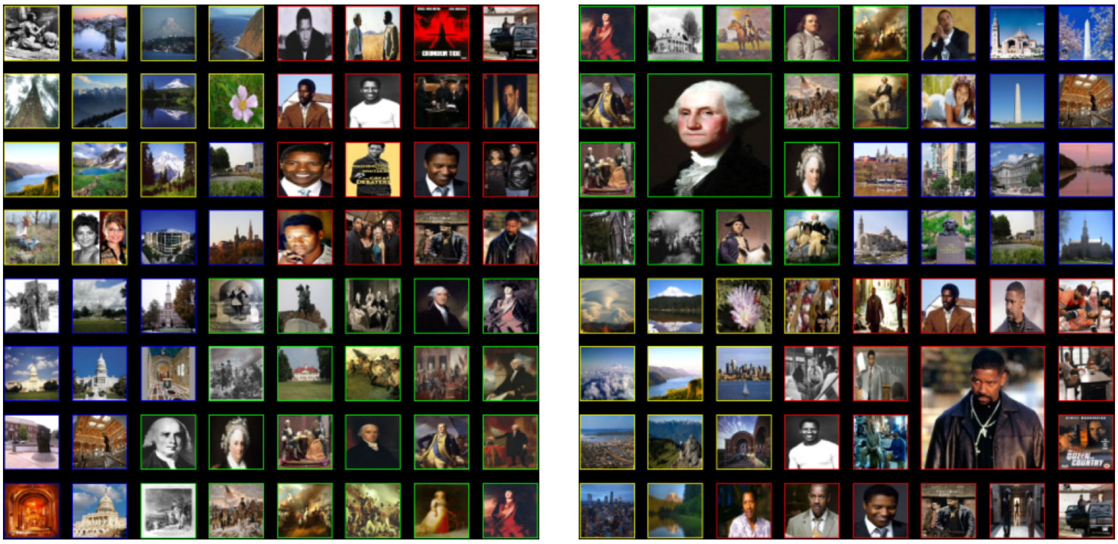


Figure 2.11: The MCM layout for the query “Washington“ in an image collection. Left is the default layout and the right side shows another layout with enlarged fixed items [SJGE13].

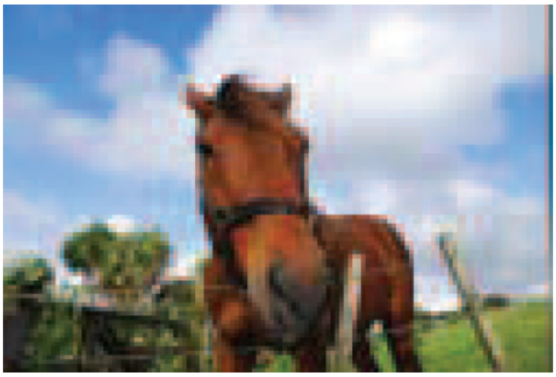
by posteriori expectation-maximation of a naive Bayesian framework with the results of a global redundancy, contrast and geometric analysis as features. To ensure an adaptive image summary to fit different target devices, and keep the proportion of the context salient regions, the authors present a grid-based piecewise linear image warping.

2.4.2 Hybrid Image Summarization

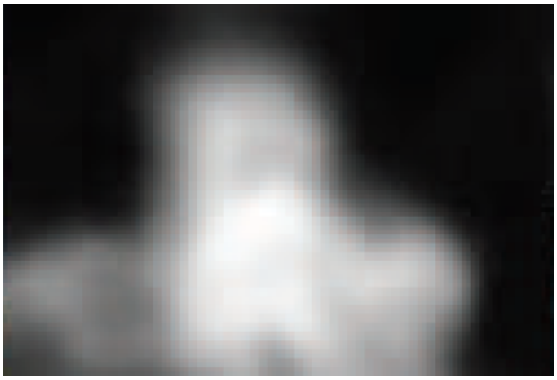
Hybrid Image Summarization [XWHL11] handles the problem of finding a few representative images to represent the set semantically and visually (see Figure 2.13). This is achieved by using a hybrid way, which uses visual and textual information associated with the images. The authors call their approach Homogeneous and Heterogeneous Message Propagation (H^2MP), because it works with homogeneous and heterogeneous relations of the images respectively tags. Since the authors use heterogeneous data too, this approach goes beyond the affinity propagation algorithm, which is only able to handle homogeneous data.

The associated information contains the following three useful relations from images and tags:

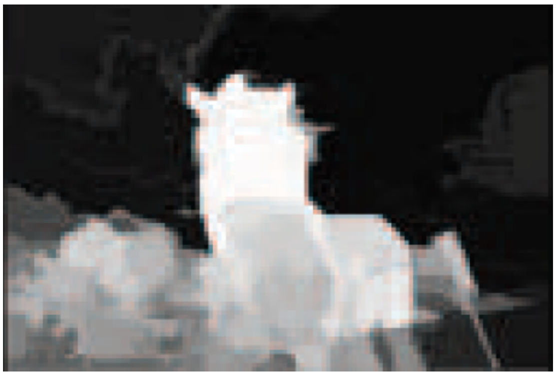
1. Homogeneous relation within images, i.e., image similarity
2. Homogeneous relation within tags, i.e., tag similarity
3. Heterogeneous relation between images and tags, i.e., their association relation



(a) Original image.



(b) Statistical saliency.



(c) Geometric constraint.

(d) Context saliency map.

Figure 2.12: Steps of the context saliency map generation [SWX$^+$09].

(a) A set of tagged images

(b) Generated summary

Figure 2.13: A summary generated from the given input by the proposed approach [XWHL11].

Especially the relation 2 and 3 are used by the authors to generate a summary that is visually and semantically satisfying. Figure 2.14 shows those relations, where $\mathcal{E}^I$ denotes the homogeneous relations within images, $\mathcal{E}^W$ the homogeneous relations within tags and $\mathcal{E}^R$ the heterogeneous relations between images and tags.

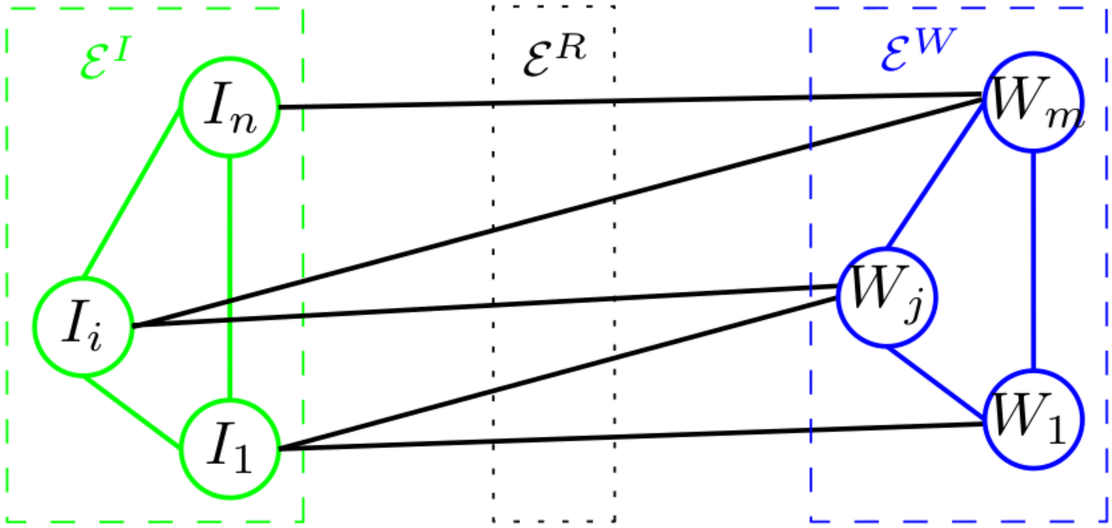


Figure 2.14: Relations between images and tags [XWHL11].

2.4.3 Image collection summarization via dictionary learning for sparse representation

Yang et al. [YSPF13] present a method for automatic image summarization by reformulating this task into a dictionary learning for sparse representation problem. Furthermore the authors discovered that they can interpret this reformulation under the Scale Invariant Feature Transform (SIFT) Bag-of-Words (BoW) framework. With this approach it is possible to approximately reconstruct each image in the original image set by weighted linear combination respectively, by the accumulation of the probabilities of various visual words of the summary images (see Figure 2.15).

2.5 Illustration of Image Search Results

This section describes the illustration of image search results by using different kind of methods. Such methods try to support the user in his search and navigation through big image datasets by using different visualization techniques. This can be achieved for example by arranging the images according to their similarity and mapping them on some 2- or 3-dimensional geometry. Another way could be to cluster the image dataset and show only some representative images per cluster. In other words, the user is able to perform a hierarchical search, whereas all the images of a specific cluster are only shown on demand.

2.5.1 Spatial Visualization for Content-Based Image Retrieval

Spatial Visualization for Content-Based Image Retrieval [MTH01] is a technique that extends traditional Content-Based Image Retrieval (CBIR), which only displays images in

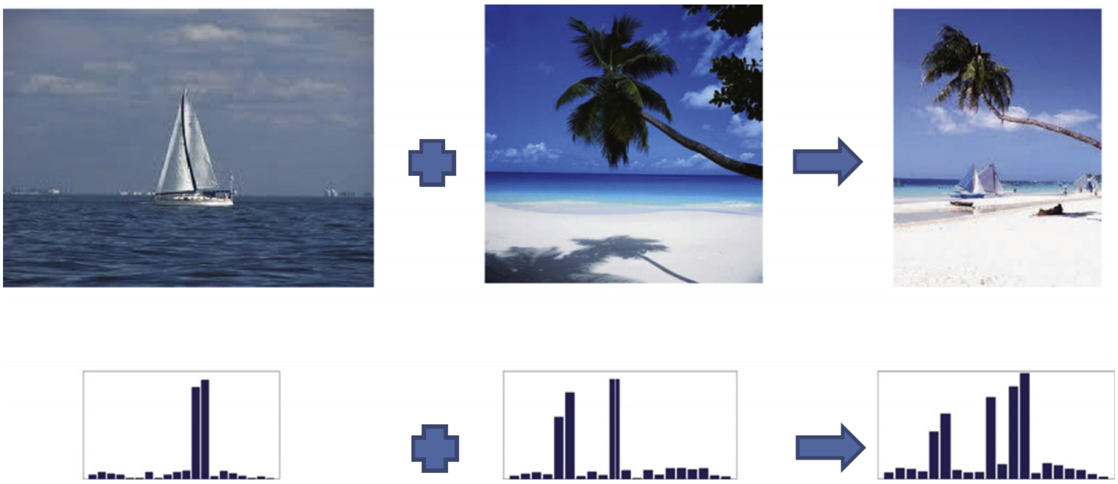


Figure 2.15: The summary images “beach“ and “palmtree“ with its reconstruction image, which has both visual objects [YSPF13].

order of decreasing similarity, which can be considered as a 1-dimensional representation. The major drawback of the traditional technique is, that relevant images can appear at separate places and the user would like to discover the relations among them. Therefore, the proposed approach extends the traditional one by ordering the images also according to their mutual similarities and visualizing them in a 2-dimensional representation. To achieve this, the authors perform a Principal Component Analysis (PCA) on the retrieved images in a first step to reduce the high-dimensional feature space to 2D, to visualize the result as a so-called *PCA Splat*. One drawback of the *PCA Splat*, is, that images can partially or totally overlap. So the authors optimize the *PCA Splat* in a second step. This is achieved by using a cost function that minimizes the overlap and deviation of the initial positions. Figure 2.16 compares the regular *PCA Splat* and the optimized one.

2.5.2 Similarity-Based Visualization for Image Browsing Revisited

The approach by Schöffmann and Ahlström [SA11] describes a simple but efficient algorithm, that sorts images based on their color similarity. It generates an intuitive arrangement of images and allows to apply it to several different layouts.

In most CBIR applications the result is represented as a list that is sorted according to its metadata. However, this is not the best way to represent the result, because users rather search by visual similarity than on metadata. So it makes sense to sort images according to their visual similarity.

The proposed algorithm has the following characteristics:

1. Fast in terms of run-time (mobile devices and adaptive interfaces possible)
2. Results in an intuitively sorted layout

(a) PCA Splat (b) Optimized PCA Splat

Figure 2.16: Comparison of regular and optimized PCA Splat [MTH01].

3. Preserves visibility of all images

The basic idea of the authors is to sort the images according to their dominant hue in the Hue Saturation Value (HSV) color space. To achieve this the algorithm consists of two steps. In the first step they classify the pixels of the images using a 16-bin hue histogram and use the index of the dominant bin as sorting criteria. In the second step they sort the images belonging to the same dominant bin again, with a different criteria. The authors use a 24-bin HSV histogram (16-,4-,4-bin) for these images and minimize the Euclidian distance between adjacent images. They also created two simple rules to filter bright and dark images, so that the algorithm produces a sequence of bright images, followed by the regular hue sorted images, finally followed by the dark images. Figure 2.17 shows the algorithm applied on the *WANG 1000* dataset [WLW01, LW03] and a 3D cylinder layout.

Finally, their user study showed that their approach improved the interactive search performance by about 20% in a common storyboard. Furthermore, the study also showed that the participants preferred the sorted storyboard over the random one, and that the participants are able to understand and efficiently use their approach.

2.5.3 Visual Diversification of Image Search Results

The work of van Leuken et al. [vLGOvZ09] introduce new methods to diversify image search results by creating a visual diverse ranking. They apply clustering on the images based on their visual characteristics. For each cluster a representative image is shown and it is also possible to explore the other images in the cluster through user interaction. Figure 2.18 shows an example clustering for the ambiguous query "jaguar", where the cluster representatives form a diverse set for the image search result.

Figure 2.17: WANG 1000 dataset sorted with the proposed approach and aligned on a 3D cylinder [SA11].

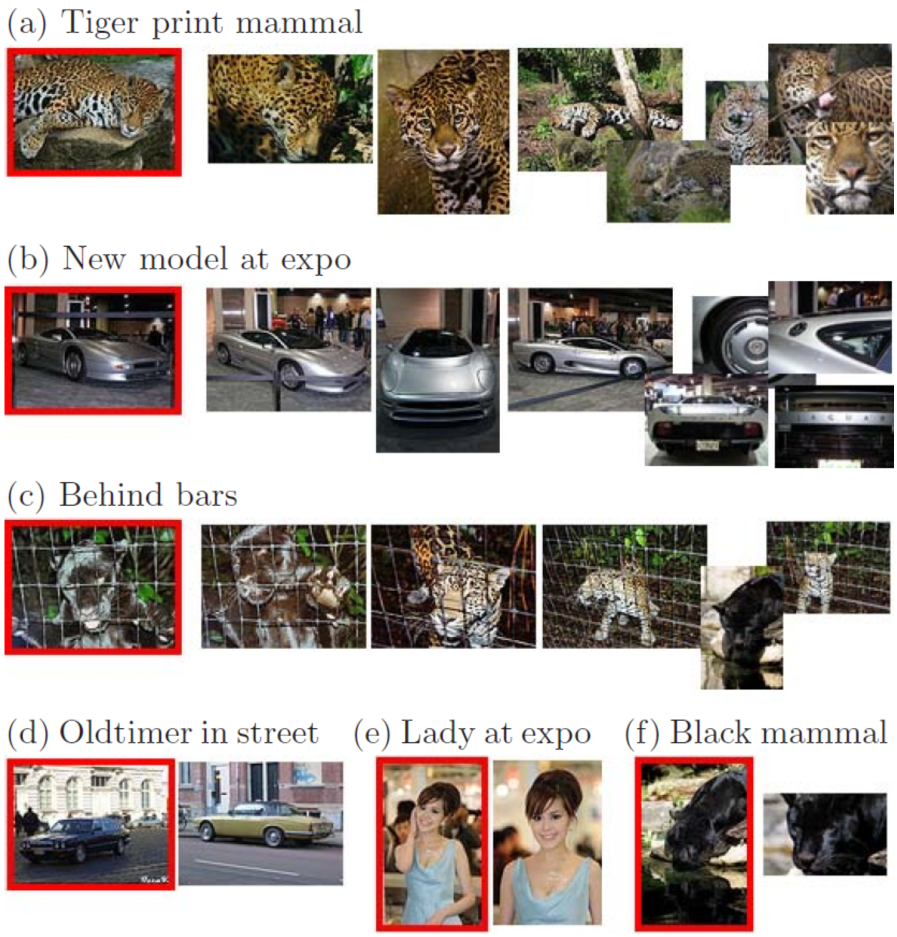


Figure 2.18: Example clustering for the ambiguous query “jaguar“. Cluster representatives are highlighted by a red border [vLGOvZ09].

CHAPTER 3

Methodology

The visualization pipeline [DSB04] is a well-known and reliable concept in information visualization. Therefore, it is used in this thesis to generate the final visualization starting from the raw data. Figure 3.1 describes the step-wise process of creating a visual representation of data.

1. The *Data Analysis* step prepares the data for the visualization, e.g. by smoothing or interpolating values, or correct some errors.

2. The *Filtering* step selects a subset of data from the before prepared data, which should be visualized.

3. The *Mapping* step uses the before filtered data and maps them on some kind of primitive geometry. It also applies some attributes like color, position and size. It is also the most critical step in the visualization pipeline for achieving expressiveness and effectiveness.

4. The *Rendering* step simply renders the before mapped data, but some effects like blending are also possible.

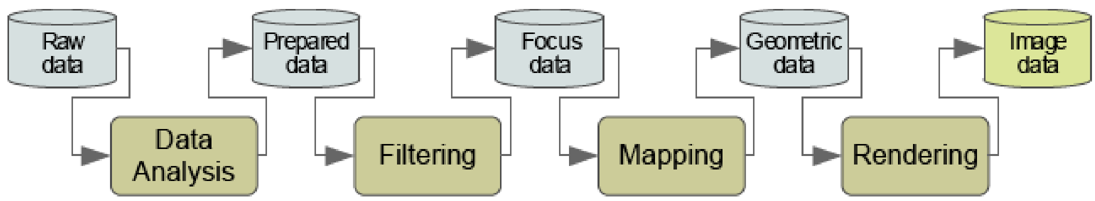


Figure 3.1: Visualization Pipeline [DSB04].

3.1 Data Analysis

The *Data Analysis* step gets the *Raw data* as input and extracts respectively derives additional data. In this work, we extract the geographical location from the Exif data and compute a saliency map based on the raw image. These two steps are explained in detail in the following subsections.

3.1.1 Geographical Location

In the Exif data the geographical location is stored as location and location reference. The longitude and latitude are stored as *Degrees*, *Minutes* and *Seconds*. The location reference indicate whether the longitude is East or West, respectively, and whether the latitude is North or South. These location references are needed to convert the geographical location to its signed version. If the reference has the value S or W the sign has to be negative, so that the resulting interval for longitude is $[-180, 180]$ and for latitude it is $[-90, 90]$. Algorithm 3.1 shows the conversion in pseudo code.

Algorithm 3.1: ExifGpsToScalar

Input: A PropertyItem *location*, and a PropertyItem *locationRef*
Output: A scalar *coordinate*

$coordinate = location.Degrees + \frac{location.Minutes}{60.0} + \frac{location.Seconds}{3600.0}$;
if $locationRef == "S" \,||\, locationRef == "W"$ **then**
  $coordinate = -coordinate$;
end
return *coordinate;*

These coordinates are defined as spherical coordinates. Since the filtering step uses *k-Means* clustering, which uses the Euclidean distance, spherical coordinates are not suitable. A conversion from spherical to cartesian coordinates is possible to avoid discontinuity. This conversion uses the trigonometric functions *cos* and *sin* applied on the longitude respectively latitude together with the mean earth radius to calculate the position on the surface as 3D cartesian coordinates. Figure 3.2 illustrates the relationship between spherical and cartesian coordinates, and Algorithm 3.2 shows the conversion from spherical to cartesian coordinates in pseudo code.

3.1.2 Saliency

We employed the work by Hou et al. [HZ07] to calculate the saliency. Their approach is implemented in *OpenCV*, but not in the C# wrapper *EmguCV*, therefore we converted this approach from C++ to C#.

One benefit of the approach of the authors is, that their model is independent of features, categories, or other forms of prior knowledge of the objects, which the image contains. It is therefore defined as a general approach which can be applied to many different data

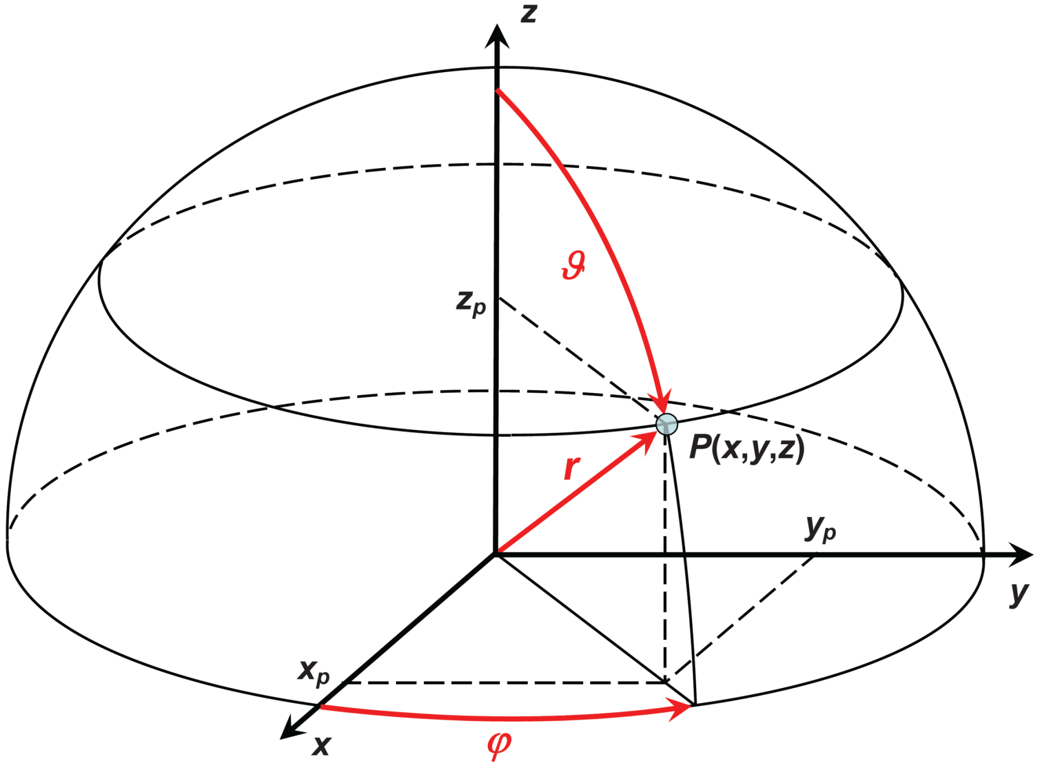


Figure 3.2: Relationship between spherical and cartesian coordinates [sph].

Algorithm 3.2: SphericalToCartesian

Input: A scalar *latitudeDegree*, and a scalar *longitudeDegree*

Output: A struct *cartesianCoordinates*

$radius = 6371.0$; `// mean earth radius in km`

$latitudeRadian = \frac{\pi}{180.0} \cdot latitudeDegree$;

$longitudeRadian = \frac{\pi}{180.0} \cdot longitudeDegree$;

$cartesianCoordinates.X = radius \cdot \cos(latitudeRadian) \cdot \cos(longitudeRadian)$;

$cartesianCoordinates.Y = radius \cdot \cos(latitudeRadian) \cdot \sin(longitudeRadian)$;

$cartesianCoordinates.Z = radius \cdot \sin(latitudeRadian)$;

return *cartesianCoordinates;*

---

sets. This is achieved by using a similar saliency detection process like in the human visual system. It is believed that two steps are involved in the visual processing. First, the parallel, fast and simple *pre-attentive* process, and second, the serial, slow and complex *attention* process. However, the authors try to simulate the behavior of the *pre-attentive* visual search, because it works with low-level features such as orientation, edges or intensities, which can indicate a candidate of an object, without further knowledge. For that to happen, the authors start with an analysis of the log spectrum of each image and obtain the spectral residual. Afterwards, they transform the spectral residual to the spatial domain to obtain the *saliency map*, which suggests the positions of the so-called *proto-objects*, which is a term to address a candidate, that has been detected but not yet identified as an object. Finally, an object map can be obtained by using a simple threshold operation. Figure 3.3 shows their approach applied to an image, which contains three objects.

In this thesis, the saliency map is used in the later mapping step to place the images properly in the *Voronoi* diagram.

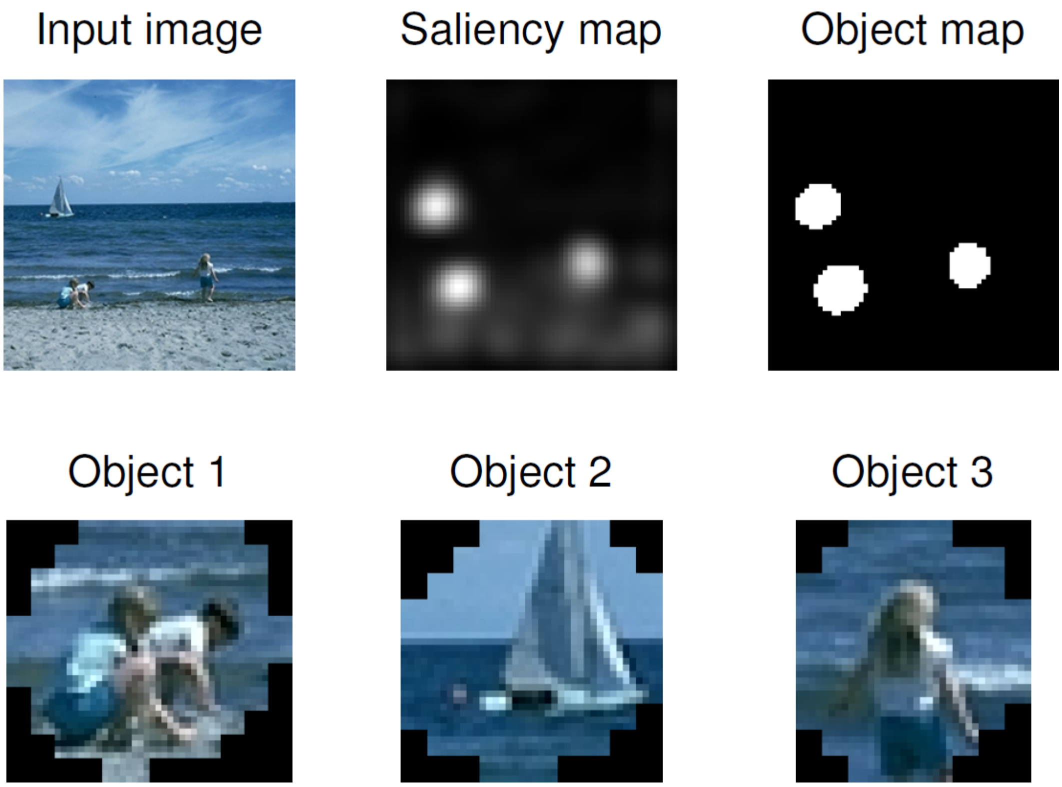


Figure 3.3: The result of the saliency detection using the spectral residual approach of Hou et al. [HZ07].

3.2 Filtering

The *Filtering* step receives the *Prepared data* from the *Data Analysis* step as input and filters it in two steps. In a first step *k-means* is used to cluster the images according to their geographical location. In a second step there are two strategies to select representative images for each cluster. These steps are explained in the following subsections.

3.2.1 Clustering

We employ the well-known *k-means* [M+67] clustering algorithm to cluster the images. A feature matrix is generated in a first step, by taking the previous calculated 3D cartesian coordinates as feature and adding them row-wise to the feature matrix. After that step *k-means* is applied with this feature matrix, a user-defined k and the *k-means++* [AV07] process for the initialization. The result is a label vector, which assigns each image to a cluster. Furthermore, a matrix containing the k 3D cluster centers is generated.

3.2.2 Selection of representative Images

There are many different possibilities how to extract a representative image from a subset of the data set. The following two simple approaches are implemented as entry point and can be selected by the user. In both cases m is a user-defined parameter and reaches from *#imagesOfSmallestCluster* up to *upperBound*.

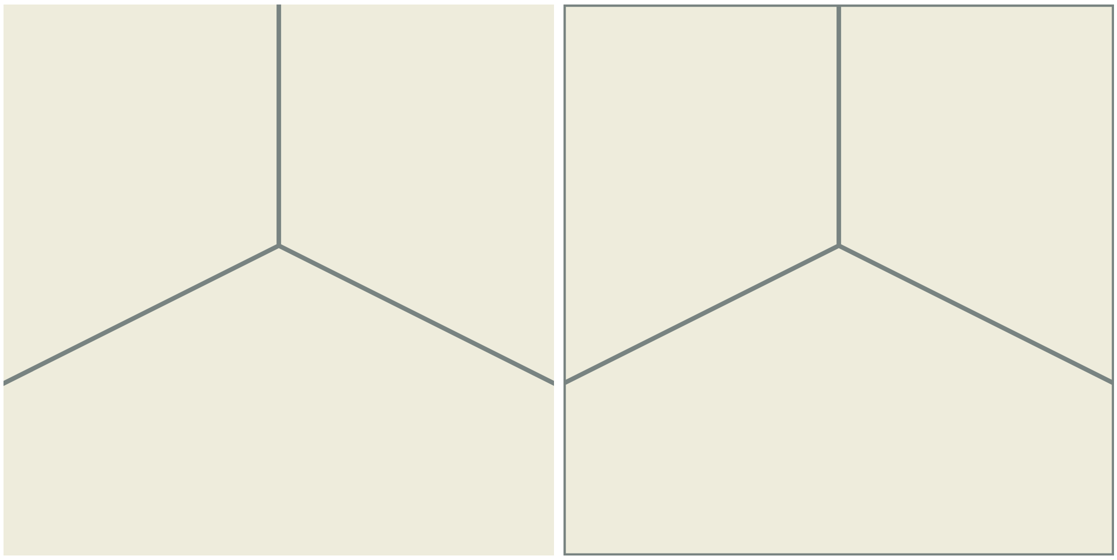
1. *Random*
   This approach simply takes m random images from a cluster, while taking care that no duplicates are generated.
2. *Minimal Distance*
   This approach calculates the distance from each image within a cluster to its cluster center. After that, the images are ordered ascending to their distance. Finally the first m images are selected.

3.3 Mapping

The *Mapping* step receives the *Focus data* from the *Filtering* step as input and maps them in two steps. In a first step a *Voronoi* diagram is generated using as many cells as clusters. In a second step the generated *Voronoi* diagram is tessellated according to the number of images within a cluster. These steps are explained in the following subsections. Furthermore, the gravity center of the saliency map is determined by using image moments. The image is then mapped, so that the gravity center of the actual image is on the same position as the gravity center of the corresponding *Voronoi* cell in the actual visualzation.

3.3.1 Voronoi Diagram

The *Voronoi* diagram is generated by providing some samples to the *Emgu CV* planar subdivision Application Programming Interface (API). The samples are a subset from a number of predefined, equal distributed 2D points, containing as much samples as clusters. Since the generated *Voronoi* diagramm may exceed the screen space it has to be clipped in a post processing step. Figure 3.4a shows the unclipped *Voronoi* diagram for three clusters with some vertices outside the screen space and Figure 3.4b shows the clipped *Voronoi* diagram, which is visualized by the border surrounding the square.



(a) The unclipped *Voronoi* diagram. (b) The clipped *Voronoi* diagram.

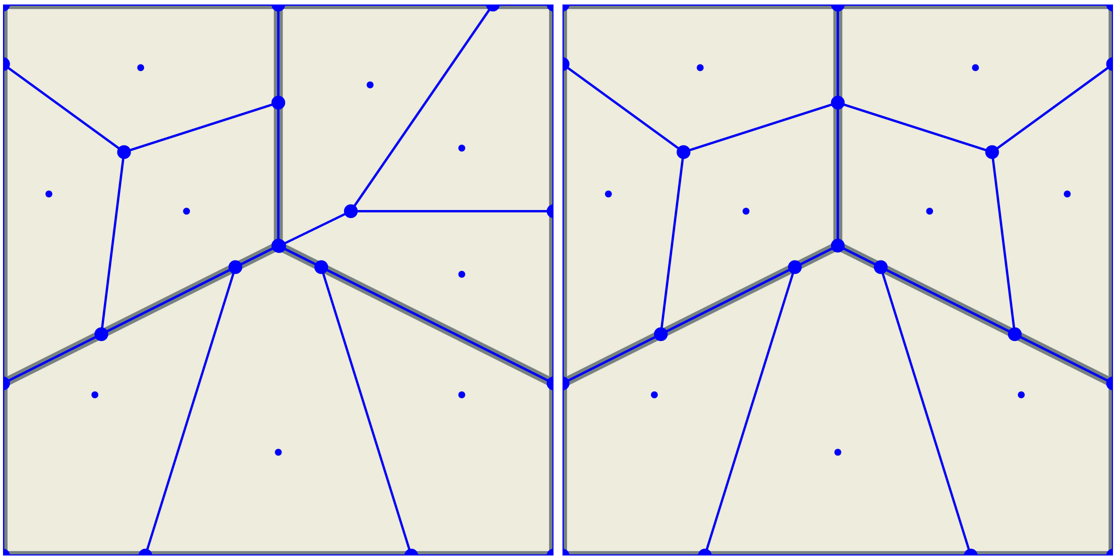
Figure 3.4: The resulting *Voronoi* diagram for three clusters.

3.3.2 Tessellation of Voronoi Diagram

In this step each *Voronoi* cell gets tessellated according to the number of images within the corresponding cluster. Algorithm 3.3 describes the basic approach of the symmetric tessellation. In the case of one image within a corresponding cluster returns the *Voronoi* cell itself as a list. In the other case, and if it is not the first *Voronoi* cell that gets processed, the algorithm checks if the current cell has a symmetric equivalent in the previous *Voronoi* cells. It checks the symmetry on the X-, Y- and XY-axis and returns a duplicate flipped on the correct axis if the condition is fulfilled. If there is no symmetry at all or it is the first *Voronoi* cell that gets processed, the algorithm continues with the regular tessellation part.

Figure 3.5 shows the comparison of the regular and the symmtetric tessellation. The different layouts for the top left and top right *Voronoi* cell in Figure 3.5a occur due to the ear clipping. Multiple solutions are available to triangulate the same or a mirrored polygon, and therefore multiple different tessellations are possible. Figure 3.5b on the other hand shows the obviously better symmetric solution for the top right *Voronoi* cell, which is achieved by simply flipping the already generated tessellation for the top left *Voroni* cell on the y-axis.

Algorithm 3.4 describes the basic approach of the regular tessellation. First the *Voronoi* cell is triangulated by using ear clipping. Afterwards the triangles are subdivided if the number of triangles is smaller than the number of images within a cluster. This subdivision takes care of the area and angles of the triangle. It subdivides the triangles with the largest area at first and always through the largest angle. After the triangulation



(a) The regular tessellation. (b) The symmetric tessellation.

Figure 3.5: Comparison of the regular and symmetric tessellation.

process it is checked if there are more triangles than images within a cluster. If this condition is true, our algorithm merges the smallest adjacent triangles to polygons until the necessary amount of polygons is reached. Finally, the centroids of the triangles (polygons) are determined and the *Voronoi* cells are generated in the same way as in section 3.3.1, by providing the centroids as samples and the current *Voronoi* cell as clipping path. Figure 3.6 shows the triangulation of a *Voronoi* diagram with three clusters and three images per cluster, where the big green points denote the vertices of the triangles and the small green points the gravity centers.

3.4 Rendering

The *Rendering* step receives the *Geometric data* from the *Mapping* step as input and renders it, which results in the *Image data*. Depending on the rendering mode, the resulting layout of the previous *Mapping* step or the actual visualization is rendered. The former allows to render the tessellation additionally to the *Voronoi* diagram and the latter has the opportunity of an optional blending step for the actual visualization. However, first of all the *Geometric data* are scaled to the user-provided resolution no matter which rendering mode is selected.

To render the *Voronoi* diagram, each of the *Voronoi* cells is drawn by a polyline along its vertices. If the tessellation option is activated, this process is repeated again for the tessellated *Voronoi* cells within a parent *Voronoi* cell.

To render the actual visualization, first of all the *Voronoi* cells have to be transformed

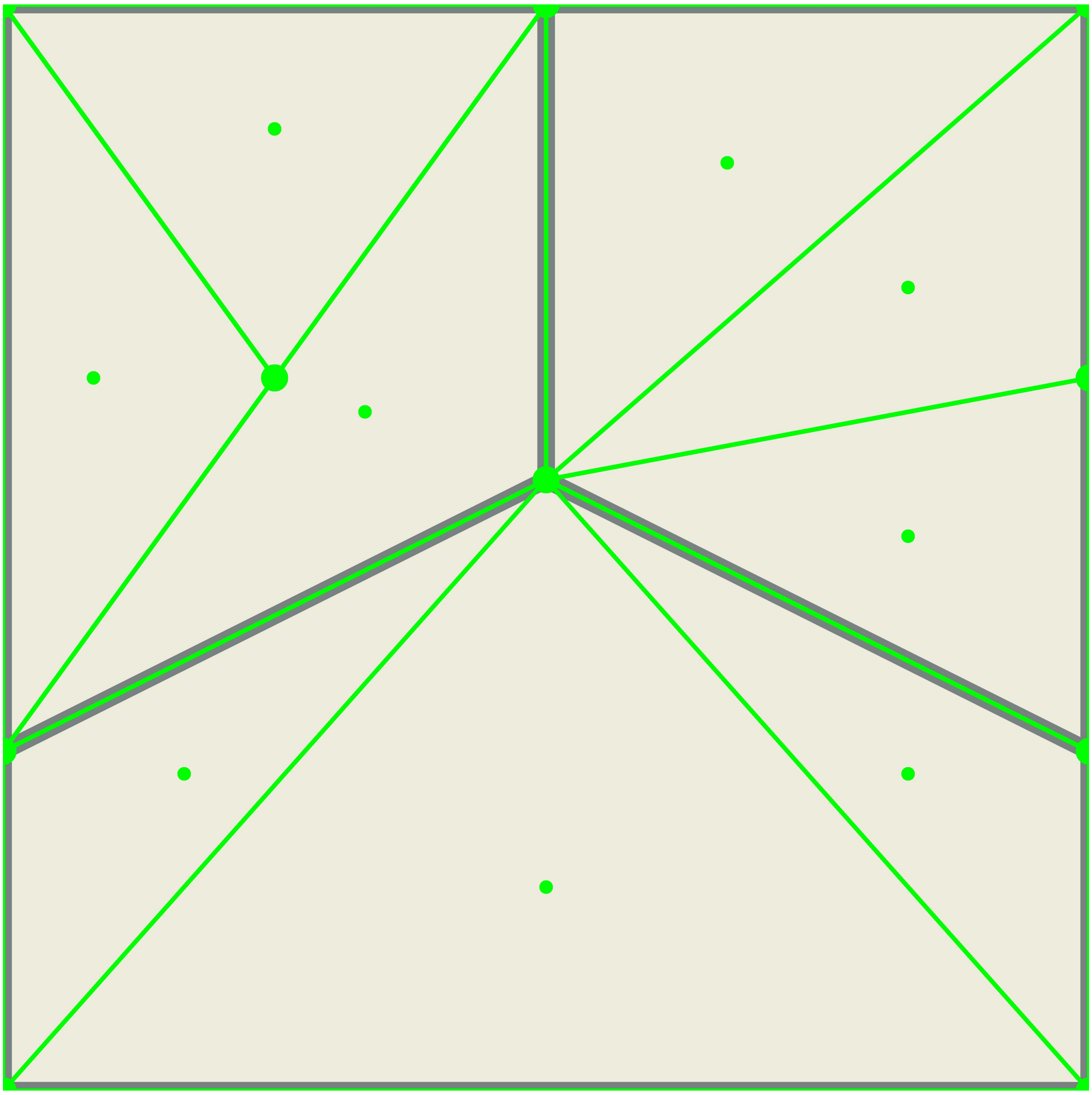


Figure 3.6: The triangulation of the three clusters of the *Voronoi* diagram into three triangles per cluster by using ear clipping and subdivision.

to the salient regions in the image space (i.e., use the gravity center of the saliency map to map the center point of the *Voronoi* cell), where the *Voronoi* cells work as a template to extract regions of interest respectively a bounding box around the *Voronoi* cell is used. This bounding box has a user-defined overlap, the so-called *blending overlap*, which is used later on to have enough information of the images to blend them along the edges. Afterwards, the image information within this bounding box is transformed to the final position in the actual visualization and rendered. Furthermore, an optional dissolve blending step is possible at this point, which is described in the next section

Algorithm 3.3: SymmetricTessellation

Input: A scalar $numberOfImages$, a VoronoiFacet *current*, and a list of VoronoiFacet *previous*
Output: A list of VoronoiFacet

```
if numberOfImages == 1 then
    // Return the current voronoi cell as list.
end
forall previous do
    if the current voronoi cell is symmetric on the X axis with a previous cell then
        // Return the tessellated voronoi cells of previous
           flipped on the X axis.
    end
    else if the current voronoi cell is symmetric on the Y axis with a previous cell
     then
        // Return the tessellated voronoi cells of previous
           flipped on the Y axis.
    end
    else if the current voronoi cell is symmetric on the XY axis with a previous
     cell then
        // Return the tessellated voronoi cells of previous
           flipped on the XY axis.
    end
end
// No previous voronoi cell is symmetric to the current one
return Tessellation(numberOfImages, current);
```

(3.4.1). Finally, after all the images were rendered, the borders of the *Voronoi* diagram are rendered as mentioned before.

3.4.1 Blending

At first glance a blending mask is generated by drawing a polyline, with the thickness of the user-defined blending overlap, along the shared edges of the tessellated *Voronoi* cells within a parent *Voronoi* cell. Afterwards the blending mask is clipped with the outline of these parent *Voronoi* cell, so that there is no dissolve blending between different clusters respectively parent *Voronoi* cells. This blending mask is than used to get the relevant pixels for the actual dissolve blending. Basically, the algorithm adds the color values of all the overlaying images per pixel to a list, and chooses the color of a randomly selected pixel of this list and continuous with the next pixel, as long as all pixels from the blending mask are processed.

Algorithm 3.4: Tessellation

Input: A scalar $numberOfImages$, and a VoronoiFacet $current$

Output: A list of VoronoiFacet

```
// Triangulates and subdivides the voronoi cell in
   numberOfImages sub-cells
triangles = Triangulate(current, numberOfImages);
if #triangles > numberOfImages then
    // Merge the smallest adjacent triangles until it fits
       numberOfImages.
end
centeroids = The centeroids of the generated triangles respectively merged
  polygons;
return CreateVoronoiFacets(centeroids, current);
```

3.5 Interaction

The well-known concept of the visualization pipeline also allows the user to interact with the process of the creation of the information visualization in every single step and to apply the changes immediately. Figure 3.7 shows the interaction possibilities for the *Filtering*, *Mapping* and *Rendering step*. Moreover, it is possible to select new *Raw data* and start the automatic *Data Analysis* step.

The *Filtering* step has the opportunity to set the quantity of clusters and the quantity of images within a cluster. Furthermore, it is possible to change the method which is used to select representative images for each cluster. It is also possible to reset the already done filtering, so that the *Prepared data* are displayed again on the world map.

The settings of the *Mapping* step allow the user to show the cluster centers of the *Focus data* in the visualization view, and to move them across the plane, which results in a new layout. The automatic update is an option to save resources, so that the new layout is computed only on demand.

The *Rendering* settings provide the opportunity to change the rendering mode, either the layout of the *Voronoi* cells, or how the actual visualization of the images is rendered. In the *Voronoi* mode it is also possible to choose if the tessellation of the *Voronoi* cells should also be rendered. Furthermore, it is possible to activate the blending for the visualization of the images and change the size of the blending overlap for the shared edges of the images. Finally, users can change the resolution of the visualization.

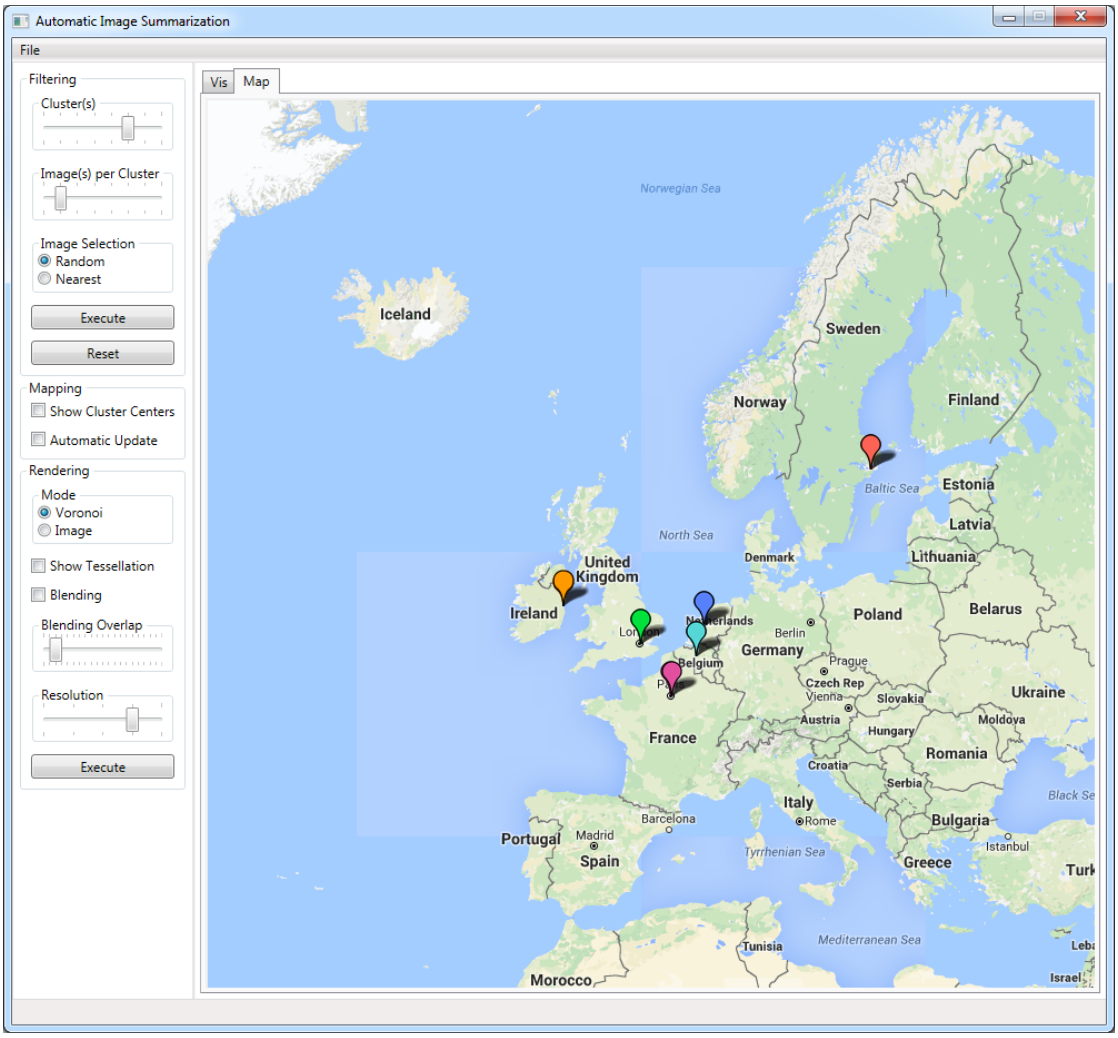
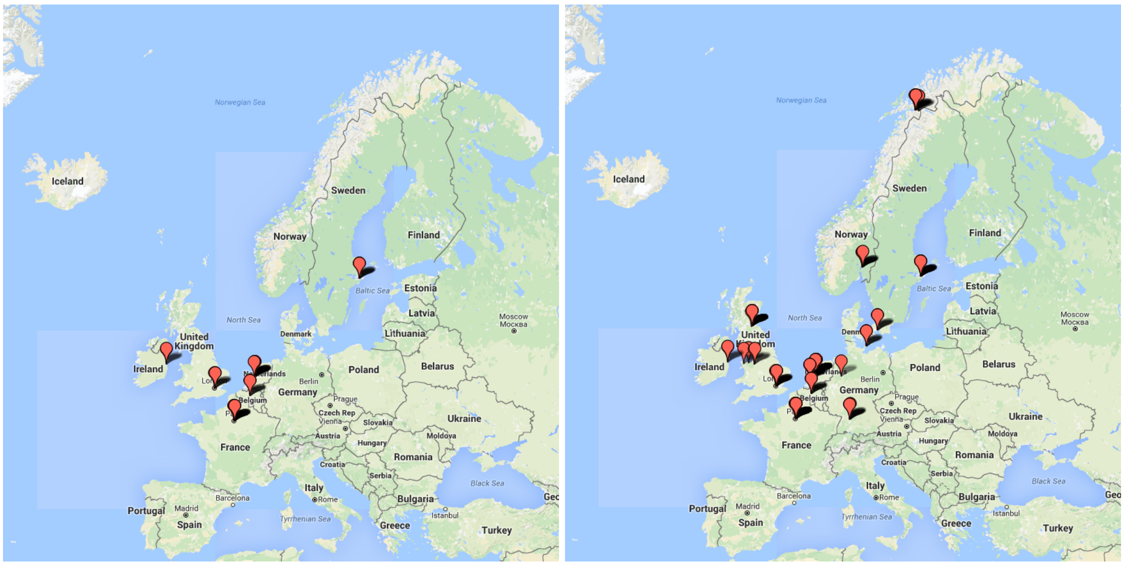


Figure 3.7: The user interface with its interaction opportunities for filtering, mapping and rendering.

CHAPTER 4

Results

In this chapter we discuss results created by employing the algorithm as explained in Chapter 3. An image dataset from an Interrail trip through Europe in 2014 serves as a demonstration case, and a subset of manually selected images from this dataset serves as a smaller data set. The full dataset contains 154 images, whereas the subset contains only 35. Figure 4.1 shows where these images are located on the world map for each dataset.

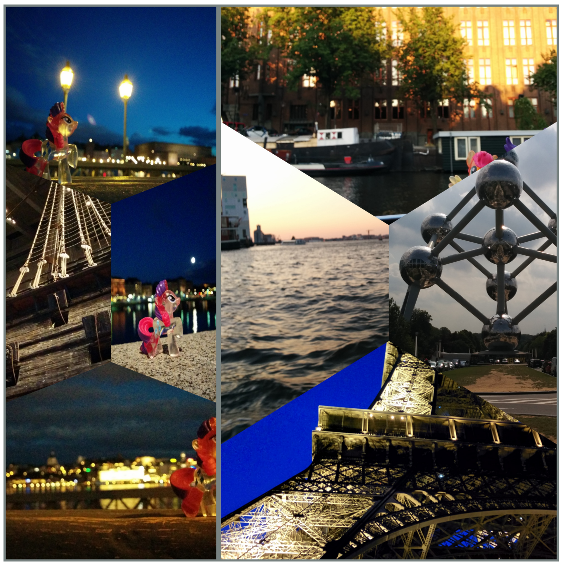


(a) The subset. (b) The full dataset.

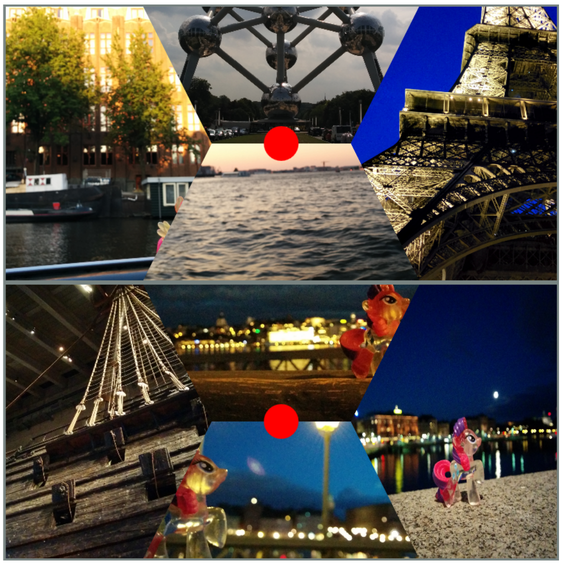
Figure 4.1: Location of the images on the world map.

Figure 4.2 shows the automatically created layout, and also a layout that was adjusted by the user. The red points in Figure 4.2c and Figure 4.2d denote the cluster centers, which are used to generate the *Voronoi Diagram*. To generate a new layout, the user is able to drag the cluster centers across the plane and a new *Voronoi Diagram* is generated in real-time. If the user drops the cluster center at its new destination, the final visualization of the image collage is rendered.

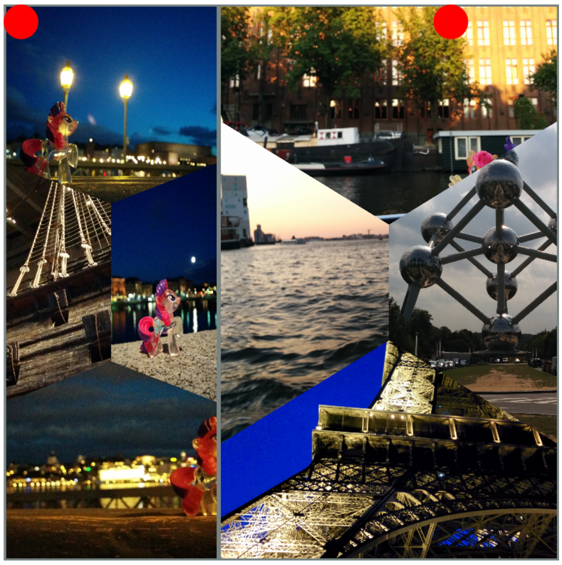
(a) The original layout.



(b) The adjusted layout.



(c) The original layout with its cluster centers.



(d) The adjusted layout with its cluster centers.

Figure 4.2: The original layout and the user adjusted layout.

The two simple image selection methods *random* and *nearest* generate diverse and appealing image collages for the tested image datasets, but for some scenarios there is a lack of diversity in the resulting image collage. In the subset, which contains 35 images, there are, for example, seven images of the Eiffel Tower and another three of the Atomium. Therefore, there is a much higher chance to get an image from the Eiffel Tower or Atomium at random. In Figure 4.3a that resulted once in three out of four images from the Eiffel Tower by using the simple *random* approach. Another lack of diversity is shown in Figure 4.3b by using the *nearest* method. This happens, because for these specific parameters with two clusters and four images per cluster, the three images of the Atomium are the nearest to its cluster center. However, the benefit of the *random* approach compared to the *nearest* is, that the *random* approach will not get stucked with the same representative images. Another problem of the *nearest* method is, the more images from a specific object are taken, the nearer the cluster center will be at the position of this specific object and therefore less images from other objects are choosen as representative images. These examples represent extreme cases and like in all other figures of this chapter, both approaches create appealing image collages and are able to select a diverse subset of representative images in general.

(a) Random images within a cluster.

(b) Nearest images to cluster center.

Figure 4.3: Lack of diversity in the image collage caused by the image selection method.

Figure 4.4 shows some results for an image collage with different blending overlap. The simple dissolve blending approach generates appealing results except for the *Voronoi* cell in the top left corner. Especially for the center *Voronoi* cell our approach produces satisfying results, because of the similarity of the images. Depending on the image dataset (i.e., the similarity of adjacent images) our approach produces satisfying summarization results. There are still directions for future work, like employing more elaborate blending techniques like *Poisson Blending* [PGB03].

The spectral residual approach for saliency detection by Hou et al. [HZ07] could be satisfyingly employed to the subset. Only the position of the image in the bottom center

(a) Blending off.

(b) Blending with 35 pixel blending overlap.

(c) Blending with 65 pixel blending overlap.

(d) Blending with 100 pixel blending overlap.

Figure 4.4: Comparision of different blending overlap.

Voronoi cell is not convincing (see Figure 4.5a). Therefore, it would be desirable to rearrange some specific images manually afterwards to improve the results. Due to the manual pre-selection with some attention to the saliency in the images, automatically generated results of the subset are more convincing than for the full dataset, which afterwards required manual tweaking to improve the visual representation (see Figure 4.5b).

Comparing the layout extreme cases, one cluster with eight images per cluster (see

(a) The subset. (b) The full dataset.

Figure 4.5: Comparison of the saliency of both datasets.

Figure 4.6a) and eight clusters with one image per cluster (see Figure 4.6b), the result of the latter are more aesthetic. The triangular layout of the image collage in Figure 4.6a is the result of the creation of the seed points for the *Voronoi Tessellation*. According to the parameter settings, the proposed approach divides the quad of the cluster into eight triangles and uses their centroids for the *Voronoi Tessellation*, which obviously results again in eight triangles. However, the resulting image collage in Figure 4.6b looks more aesthetic, because the seed points for the *Voronoi Tessellation* have been pre-defined by the user, with some respect to a an equally distributed appearance.

The visualization might produce erroneous results due to misleading symmetry. Figure 4.7 shows two examples with four clusters and an extra misleading *pseudo* cluster in the center part of the image collage. This *pseudo* cluster is caused by the pre-attentive processing of the human visual system, which works with low-level features such as orientation, edges or intensities. Depending on the similarity of the images in this *pseudo* cluster, the effect is more or less visible and therefore misleading, especially at first glance. This problem occurs because the proposed approach applies the *Voronoi Tessellation* on the first cluster. For all the following clusters, it mirrors the already performed *Voronoi Tessellation* of a cluster, if there is any kind of symmetry of the already tessellated clusters and the new one. Therefore, four clusters with four images per cluster is the absolut extreme case, because there is symmetry on all axis, and also the tessellation of the first cluster is symmetric itself (see Figure 4.7b).

Figure 4.8 shows a representative image collage for the subset of the Interrail trip. For every of the six visited countries, it shows the landmark of every country, like the Eiffel

(a) One cluster and eight images per cluster. (b) Eight clusters and one image per cluster.

Figure 4.6: Comparision of two layout extreme cases.

(a) Four clusters and three images per cluster. (b) Four clusters and three images per cluster.

Figure 4.7: Some examples of misleading symmetry in the center part of the image collage.

Tower for France, the Atomium for Belgium, and the canals for Amsterdam with its houseboats, and a typical Irish breakfast. Therefore, the visualization represents a concise summary of the six visited countries, created out of a subset of 35 images.

(a) One representative image per cluster.

(b) Two representative images per cluster.

Figure 4.8: Representative image collage for the subset of the Interrail trip. The visited countries are: France, Sweden, Netherlands, Ireland, United Kingdom and Belgium.

CHAPTER 5

Conclusion and Future Work

The aim of this thesis was to implement an algorithm similar to *ImageHive* [TSLX11], which should automatically summarize a given dataset of geotagged images and creates a representative image collage. Furthermore, this thesis was focused on the useful arrangement of the images and the optimal presentation of the relevant information rather than on artistic merits.

The problem statement allows for a lot of freedom for individual solutions to visualize a representative subset of images from a given image dataset. Also, the task of visualization is rather a subjective one than an objective one, because every user has personal preferences and therefore, no ideal solution can be defined for this task. However, the results show that this task has been fulfilled with this thesis. Moreover, the mapping algorithm as proposed in this thesis has the advantage that it takes care of the symmetry of the *Voronoi* cells. The objective part is concluded with this thesis and the subjective part can be verified in the future work in a user study, after the following improvements are implemented.

For future work, we would like to improve the dissolve blending by using other kinds of blending techniques, such as color blending in *ImageHive* [TSLX11], or *Poisson Blending* [PGB03] in *AutoCollage* [RBHB06]. Another way to improve the blending result would be to create a better arrangement for the images within a cluster. Techniques like *Kernelized Sorting* [QSS09] could help to arrange images with similar colors next to each other. Through this new layout, smoother transitions between the images would be possible for each of the before mentioned blending techniques. Furthermore, we would like to improve the selection of representative images for a cluster by using the Shared Nearest Neighbors (SNN) [MHP08] approach by van Leuken et al. [vLGOvZ09], to get a diverse set of representative images. Finally, more user interaction, like dynamically scaling, and rearranging or replacing of particular images, is desirable to achieve even better results.

List of Figures

List of Algorithms

Acronyms

API Application Programming Interface. 27

BoW Bag-of-Words. 18

CBIR Content-Based Image Retrieval. 18, 19

CVT centroidal Voronoi tessellation. 8, 9, 14, 45

Exif Exchangeable image file format. 2, 24

H^2MP Homogeneous and Heterogeneous Message Propagation. 16

HSV Hue Saturation Value. 20

MCM Max Correlation Map. 15, 16, 45

MCMC Markov chain Monte Carlo. 12

PCA Principal Component Analysis. 19

SIFT Scale Invariant Feature Transform. 18

SNN Shared Nearest Neighbors. 43

WCSS within-cluster sum of squares. 7

Bibliography

[AV07] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[CBGS16] John H Conway, Heidi Burgiel, and Chaim Goodman-Strauss. *The symmetries of things*. CRC Press, 2016.

[CCS12] Hsinchun Chen, Roger HL Chiang, and Veda C Storey. Business intelligence and analytics: From big data to big impact. *MIS quarterly*, 36(4):1165–1188, 2012.

[Cox73] Harold Scott Macdonald Coxeter. *Regular Polytopes: 3d Ed.* Dover, 1973.

[Del34] Boris Delaunay. Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2, 1934.

[DFG99] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM review*, 41(4):637–676, 1999.

[Dir50] G Lejeune Dirichlet. Über die reduction der positiven quadratischen formen mit drei unbestimmten ganzen zahlen. *Journal für die reine und angewandte Mathematik*, 40:209–227, 1850.

[DSB04] Selan Dos Santos and Ken Brodlie. Gaining understanding of multivariate and multidimensional data through visualization. *Computers & Graphics*, 28(3):311–325, 2004.

[GTZM10] Stas Goferman, Ayellet Tal, and Lihi Zelnik-Manor. Puzzle-like collage. In *Computer Graphics Forum*, volume 29, pages 459–468. Wiley Online Library, 2010.

[Gul97] Jan Gullberg. *Mathematics: from the birth of numbers*. WW Norton & Company, 1997.

[HL11] Martin Hilbert and Priscila López. The world's technological capacity to store, communicate, and compute information. *Science*, 332(6025):60–65, 2011.

[HZ07] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[Llo82] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[LW03] Jia Li and James Ze Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Transactions on pattern analysis and machine intelligence*, 25(9):1075–1088, 2003.

[M+67] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA., 1967.

[MCB+11] James Manyika, Michael Chui, Brad Brown, Jacques Bughin, Richard Dobbs, Charles Roxburgh, and Angela H Byers. Big data: The next frontier for innovation, competition, and productivity. 2011.

[MHP08] Pierre-Alain Moëllic, Jean-Emmanuel Haugeard, and Guillaume Pitel. Image clustering based on a shared nearest neighbors approach for tagged collections. In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, pages 269–278. ACM, 2008.

[MTH01] Baback Moghaddam, Qi Tian, and Thomas S Huang. Spatial visualization for content-based image retrieval. In *ICME*, page 42. IEEE, 2001.

[PGB03] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 313–318. ACM, 2003.

[PSPT14] Tobias Pietzsch, Stephan Saalfeld, Stephan Preibisch, and Pavel Tomancak. Bigdataviewer: Interactive visualization and image processing for terabyte data sets. *arXiv preprint arXiv:1412.0488*, 2014.

[QSS09] Novi Quadrianto, Le Song, and Alex J. Smola. Kernelized sorting. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1289–1296. Curran Associates, Inc., 2009.

[RBHB06] Carsten Rother, Lucas Bordeaux, Youssef Hamadi, and Andrew Blake. Autocollage. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 847–852. ACM, 2006.

[RRS13] Bernhard Reinert, Tobias Ritschel, and Hans-Peter Seidel. Interactive by-example design of artistic packing layouts. *ACM Transactions on Graphics (TOG)*, 32(6):218, 2013.

[SA11] Klaus Schoeffmann and David Ahlström. Similarity-based visualization for image browsing revisited. In *Proceedings of the 2011 IEEE International Symposium on Multimedia (ISM)*, pages 422–427. IEEE, 2011.

[SEZ13] Gerald Schaefer, David Edmundson, and Shao Ying Zhu. *Online Visualisation of Google Images Results*, pages 221–230. Springer International Publishing, Cham, 2013.

[SJGE13] Grant Strong, Rune Jensen, Minglun Gong, and Anne C Elster. Organizing visual data in structured layout by maximizing similarity-proximity correlation. In *Advances in Visual Computing*, pages 703–713. Springer, 2013.

[sph] Spherical coordinates. `http://www.seos-project.eu/modules/laser-rs/images/coordinates-spherical.png`. Accessed: 2016-07-24.

[SWX+09] Liang Shi, Jinqiao Wang, Lei Xu, Hanqing Lu, and Changsheng Xu. Context saliency based image summarization. In *Proceedings of the 2009 IEEE International Conference on Multimedia and Expo (ICME)*, pages 270–273. IEEE, 2009.

[Thi11] Alfred H Thiessen. Precipitation averages for large areas. *Monthly weather review*, 39(7):1082–1089, 1911.

[TSLX11] Li Tan, Yangqiu Song, Shixia Liu, and Lexing Xie. Imagehive: Interactive content-aware image summarization. *IEEE Computer Graphics and Applications*, (1):46–55, 2011.

[vLGOvZ09] Reinier H van Leuken, Lluis Garcia, Ximena Olivares, and Roelof van Zwol. Visual diversification of image search results. In *Proceedings of the 18th International Conference on World Wide Web*, pages 341–350. ACM, 2009.

[Vor08] Georges Voronoï. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les parallélloèdres primitifs. *Journal für die reine und angewandte Mathematik*, 134:198–287, 1908.

[Wika] Wikipedia. Centroidal voronoi tessellation. `https://en.wikipedia.org/wiki/Centroidal_Voronoi_tessellation`. Accessed: 2017-04-18.

[Wikb] Wikipedia. K-means clustering. `https://en.wikipedia.org/wiki/K-means_clustering`. Accessed: 2017-04-18.

[Wikc] Wikipedia. Voronoi diagram. `https://en.wikipedia.org/wiki/Voronoi_diagram`. Accessed: 2017-04-18.

[WLW01] James Ze Wang, Jia Li, and Gio Wiederhold. Simplicity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on pattern analysis and machine intelligence*, 23(9):947–963, 2001.

[WQS+06] Jingdong Wang, Long Quan, Jian Sun, Xiaoou Tang, and Heung-Yeung Shum. Picture collage. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 347–354. IEEE, 2006.

[XWHL11] Hao Xu, Jingdong Wang, Xian-Sheng Hua, and Shipeng Li. Hybrid image summarization. In *Proceedings of the 19th ACM International Conference on Multimedia*, pages 1217–1220. ACM, 2011.

[YSPF13] Chunlei Yang, Jialie Shen, Jinye Peng, and Jianping Fan. Image collection summarization via dictionary learning for sparse representation. *Pattern Recognition*, 46(3):948–961, 2013.