Dipl.-Ing. Marco A. Hudelist, Bakk. techn.

Improving Mobile Interaction for Image & Video Search

DISSERTATION

submitted in fulfilment of the requirements for the degree of

Doktor der Technischen Wissenschaften

Alpen-Adria-Universität Klagenfurt

Fakultät für Technische Wissenschaften

Mentor
O.Univ.-Prof. Dipl.-Ing. Dr. László Böszörményi
Alpen-Adria-Universität Klagenfurt
Institute of Information Technology

1st Evaluator
O.Univ.-Prof. Dipl.-Ing. Dr. László Böszörményi
Alpen-Adria-Universität Klagenfurt
Institute of Information Technology

2nd Evaluator
Prof. Dr. Max Mühlhäuser
Technische Universität Darmstadt
Informatics Department - Telecooperation Group

Klagenfurt, June/2015

Affidavit

I hereby declare in lieu of an oath that

- the submitted academic paper is entirely my own work and that no auxiliary materials have been used other than those indicated,
- I have fully disclosed all assistance received from third parties during the process of writing the paper, including any significant advice from supervisors,
- any contents taken from the works of third parties or my own works that have been included either literally or in spirit have been appropriately marked and the respective source of the information has been clearly identified with precise bibliographical references (e.g. in footnotes),
- to date, I have not submitted this paper to an examining authority either in Austria or abroad and that
- the digital version of the paper submitted for the purpose of plagiarism assessment is fully consistent with the printed version.

I am aware that a declaration contrary to the facts will have legal consequences.

Signature: _____

Klagenfurt, 14. Juni 2015

Eidesstattliche Erklärung

Ich versichere an Eides statt, dass ich

- die eingereichte wissenschaftliche Arbeit selbstständig verfasst und andere als die angegebenen Hilfsmittel nicht benutzt habe,
- die während des Arbeitsvorganges von dritter Seite erfahrene Unterstützung, einschließlich signifikanter Betreuungshinweise, vollständig offengelegt habe,
- die Inhalte, die ich aus Werken Dritter oder eigenen Werken wortwörtlich oder sinngemäß übernommen habe, in geeigneter Form gekennzeichnet und den Ursprung der Information durch möglichst exakte Quellenangaben (z.B. in Fußnoten) ersichtlich gemacht habe,
- die Arbeit bisher weder im Inland noch im Ausland einer Prüfungsbehörde vorgelegt habe und
- zur Plagiatskontrolle eine digitale Version der Arbeit eingereicht habe, die mit der gedruckten Version übereinstimmt.

Ich bin mir bewusst, dass eine tatsachenwidrige Erklärung rechtliche Folgen haben wird.

Unterschrift: _____

Klagenfurt, 14. Juni 2015

This dedication is not available in your country. :-/

Contents

List of Tables

List of Figures

Acknowledgements

I am deeply thankful to my advisor and good friend Klaus Schöffmann. He gave me the opportunity to work on this topic and accompanied me from the days of my master thesis to my dissertation. Without him this thesis would not have been possible.

I am also very thankful to Prof. László Böszörményi for his guidance, support and different viewpoints that helped without a doubt to continue the work when I was stuck.

Furthermore, I have to thank my colleagues that gave me a helping hand when I fell into a pitfall where I could not get out on myself. Especially the discussions with my office-mate Claudiu Cobârzan were often helpful, despite their sometimes great intensities.

Moreover, I want to thank Prof. Max Mühlhäuser for agreeing to evaluate my thesis.

Additional thanks also go to Alpen-Adria-Universität Klagenfurt for enabling me to work on this thesis.

Last but not less important, I thank my family, especially my mum. Without her support I would not be able to write these lines.

The work performed in the NGVB project was funded by the Federal Ministry for Transport, Innovation and Technology (bmvit) and Austrian Science Fund (FWF): TRP 273-N15 and the European Regional Development Fund and the Carinthian Economic Promotion Fund (KWF), supported by Lakeside Labs GmbH, Klagenfurt, Austria.

Refereed Publications

(Content appears in this thesis with permission from ACM under the ACM Author Rights and Publishing Policy §2.5, Version 8)

- Marco A. Hudelist. Next Generation Image and Video Browsing on Mobile Devices. In *Proceedings of the 3rd ACM International Conference on Multimedia Retrieval (ICMR '13)*. ACM, New York, USA. (Chapters 4, 5 and 8)
- Marco A. Hudelist, Klaus Schoeffmann, Laszlo Boeszoermenyi. Mobile Video Browsing with a 3D Filmstrip. In *Proceedings of the 3rd ACM International conference on multimedia retrieval (ICMR '13)*. ACM, New York, USA. (Chapter 8)
- Marco A. Hudelist, Klaus Schoeffmann, and Laszlo Boeszoermenyi. Mobile Video Browsing with the ThumbBrowser. In *Proceedings of the 21th ACM International Conference on Multimedia (MM '13)*. ACM, Barcelona, Spain. (Chapter 8)
- Marco A. Hudelist, Claudiu Cobârzan, and Klaus Schoeffmann. OpenCV Performance Measurements on Mobile Devices. In *Proceedings of the 4th ACM International Conference on Multimedia Retrieval (ICMR '14)*. ACM, Glasgow, UK. (Chapter 7)
- Marco A. Hudelist, Klaus Schoeffmann, and David Ahlström. Evaluating Alternatives to the 2D Grid Interface for Mobile Image Browsing. *International Journal of Semantic Computing (IJSC)*, Vol. 8, Nr. 2, 185-208. World Scientific, 2014. (Chapters 4 and 5)
- Marco A. Hudelist, Klaus Schoeffmann, and Quing Xu. Improving Interactive Known-Item Search in Video with the Keyframe Navigation Tree. In *Proceedings of the 2015 International Conference of MultiMedia Modeling (MMM '15)*. Springer, Sydney, Australia. (Chapter 6)
- Claudiu Cobârzan, Marco A. Hudelist, Klaus Schoeffmann and Manfred J. Primus. Mobile Image Analysis: Android vs. iOS. In *Proceedings of the 2015 International Conference of MultiMedia Modeling (MMM '15)*. Springer, Sydney, Australia. (Chapter 7)
- Marco A. Hudelist and Qing Xu. Multi-Stripe Video Browser for Tablets. In *Proceedings of the 2015 Video Search Showcase (VSS) at the International Conference of MultiMedia Modeling (MMM '15)*. Springer, Sydney, Australia. (Chapter 8)
- Marco A. Hudelist, Klaus Schoeffmann, David Ahlström and Mathias Lux. How Many, What and Why? Visual Media Statistics on Smartphones and Tablets. In *2015 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*. IEEE, Turin, Italy. To be published. (Chapter 3)

Abstract

One of the most important functions of smartphones and tablets is capturing, storing and viewing images and videos. Never before was it so easy and effortless to take a picture. As a result, image collections grow increasingly on mobile devices. However, their current browsing interfaces are not designed to support users handling large collections in an efficient way. It is hard to find images again when users do not exactly know where the photo was filed. Users end up scrolling endlessly. The same can be said for video collections or even navigation inside a video. Discovering important scenes without knowing the exact time code is hard.

The improvement of this situation is the focus of this thesis. In a first step, the status quo is evaluated regarding mobile media usage on smartphones and tablets by performing a survey with 215 participants. In the next step, the implications of the survey influence the development of a variety of mobile 2D and 3D image and video browsing interfaces. To enable these approaches, results of content analysis are utilized. In case of images, the dominant color is used for sorting purposes. Moreover, videos are segmented with the help of a new sub-shot based approach. Three user studies have been performed - two for image browsing and one for video browsing. They show an important difference between browsing on small vs. large touchscreens and prove the utility of sub-shot visualization in a mobile setting. Furthermore, additional mobile video browsing interfaces are introduced. Finally, as content analysis is extensively utilized, performance of current smartphones and tablets regarding well-known OpenCV functions is measured, listed and discussed.

CHAPTER

1 Introduction

The 22nd November of the year 1826 marks a very important point in time for the history of images. On this date, Joseph Nicéphore Niépce was able to produce a durable and light-resistant representation of an motive captured by a camera for the first time: the view out of his workroom in Le Gras (see Figure 1.1). It took him eight hours to capture this image. Several years later Eadweard Muybridge created with a setup of several still cameras the first *moving picture* of Sallie Gardner, a horse, as can be seen in Figure 1.2. Technology advanced in fast pace, from large and expensive professional devices to smaller and cheaper personal cameras.

Figure 1.1: Oldest surviving camera photograph by J. N. Niépce.

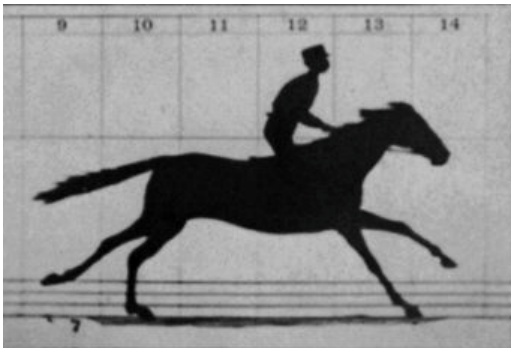

Figure 1.2: First movie captured in real-time by E. Muybridge.

With the technology also the produced content advanced: from poor black and white photographs to high quality images and movies. As capturing became easier and more affordable, also the amount of images and videos owned by an individual grew. Shelves of photo albums and film cartridges were produced. Digital cameras and camcorders were created that replaced their analog predecessors and quickly surpassed them in terms of quality and storage capabilities.

Today we live in a world were mobile phones have fused with digital cameras and mobile computers, creating the *smartphone*. In 2015 smartphones are ubiquitous. Projected smartphone sales are still rising (see Figure 1.3) and it also infused new life into another device category that never really gained momentum until reimagined during the smartphone boom: the tablet. Rather than shrinking a laptop computer down, the new generation of tablets evolved from the smartphone, adopting the user experience and capabilities. Also shipments for this device category are expected to further increase until 2017 [IDC15] (see Figure 1.4).

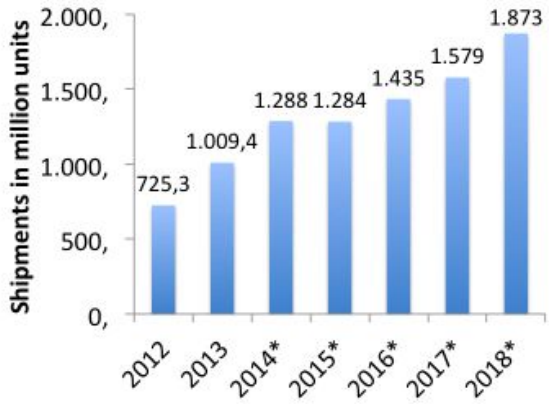

Figure 1.3: Global smartphone shipments 2012 to 2018 (* = forecast)[IDC15].

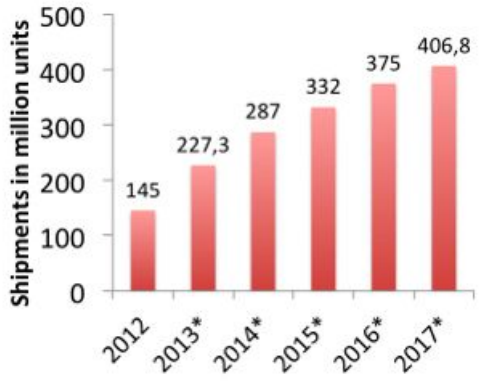

Figure 1.4: Global tablet shipments 2012 to 2017 (* = forecast)[IDC15].

1.1 The Challenge

When looking at the most popular cameras of the photo sharing site Flickr the first five places are all taken by smartphones, as can be seen in Figure 1.5. Newest smartphones offer good cameras and have the advantage that people do not have to carry around an additional device for photo capture. They have replaced traditional point-and-shoot cameras for many people. The same is true for video capture. Not too long ago you needed a special device for recording personal videos: a *camcorder*. Smartphones took over this category too. Moreover, as tablets continue to become more and more common, people use them for image and video capture too.

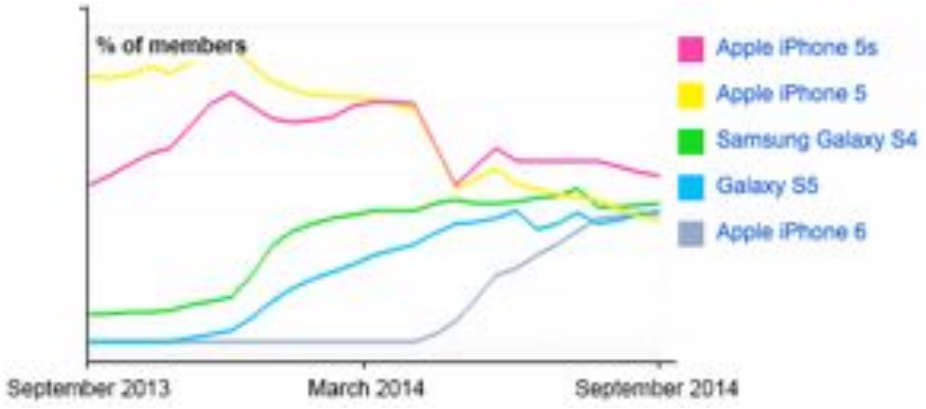

Figure 1.5: Most popular cameras in the Flickr community (adapted [Fli14]).

However, this also creates a problem: the easier it is to produce content the more content accumulates over time, as people generally tend not to delete old photos and videos. This results in large collections of media. A large collection in itself is not something bad, but users are not very likely to organize their media collections or describe the content of their media by adding tags. With such preconditions it becomes very hard to find anything although users have a clear memory about the item that they are looking for. In research such situations are called Known-Item Search (KIS) scenarios. Traditionally, image and video retrieval techniques are proposed to solve such challenges and extensive research has

been performed in that direction. Unfortunately, there is a catch to it: retrieval methods require users to be able to formulate their search request in a query - a requirement that many users are not able to fulfill. It is often too difficult for them to appropriately describe their mental image in such a way. Also the *semantic gap* is of concern as retrieval tools are still not able to decipher actual meaning and purpose of images and videos.

As a result users nowadays are still using the limited image and video interfaces created decades ago. When we look at current image browsing interfaces we are faced most of the time with simple lists or grids. Lists or grids are a good way to provide access to a small amount of items. It becomes problematic when the number of items grows. At a given time only a subset of items is visible, which entails more and more scrolling. Furthermore, this visualization makes it hard to see any content structure in the collection. What makes it worse is that when users reach the end of a list or grid they tend to first scroll all they way up again to start over [SC13]. The task becomes cumbersome, time consuming and in the end frustrating.

Similarly, current video tools seem to be stuck in the old VCR-days. Play/pause, fast forward, fast rewind and a seeker bar are the basic features that are currently available when users play a video. Imagine you want to show your best friend a funny scene in the latest episode of *Big Bang Theory* but you cannot remember the exact time code. Imagine you have the task of seeking through security cam footage to find related sequences. You have to seek through either the whole video or count on your luck and give the seeker bar a try! Sometimes it works; often you are frustrated by investing a lot of time and not finding anything.

What makes things even worse is that the same interaction techniques of PCs were transferred without any notable changes directly to smartphones and tablets. However, they were designed with keyboard and mouse in mind - not touchscreens. Due to the smaller screens even less screen space is available for large image grids and video seeker bars. On the other hand, todays increased interaction and visualization opportunities are ignored for the most part.

1.2 Finding a Better Way

The research community starts to recognize that although retrieval tools can produce impressing results, the lack of user involvement is bothersome. Worring et al. [WSS+12] said it best when they ask the question: *where is the user in multimedia retrieval?*. They request that researchers should focus more on interactive search, such as image and video browsing, as it is a user-focused way to:

- Find a specific item/sequence in a collection/video.
- Understand the structure of formerly unknown collections and videos.
- Filter collections/videos for new interesting items or sequences.

Instead of relying exclusively on the capabilities of automatic retrieval systems, browsing systems include the user as an active part in the search process. They take into account that a human being is still the best one to decide whether something is relevant or not. The system on the other hand tries to give the user enough information and overview to make efficient decisions to reach the goal as fast as possible. Furthermore, in certain scenarios it can make the process enjoyable and playful, especially when exploring new and unknown image collections or videos.

As discussed above, smartphones and tablets play an important role in image/video capture, consumption and storage. It is therefore necessary to not only perform research on image/video browsing in general but tailored to these new mobile environments. They differ in many ways from traditional computing:

- Interaction happens via touch gestures with one or multiple fingers or by changing the physical orientation of the device.
- They have powerful CPUs and GPUs capable of rich visualizations, but still have their limits due to their small and mobile nature (heat and battery constraints).
- Compared to PCs their screens are much smaller, requiring intelligent utilization of the available space.

- They are most of the time connected with a widespread set of connectivity options.
- They offer a variety of sensors to perceive their environment.

Therefore, great opportunity exists in utilizing those features and to discover better ways of image and video browsing, especially on these device categories.

1.3 Contributions

The research in this thesis investigates several approaches on how to improve image and video browsing on smartphones and tablets, and investigates several connected sub-areas that are related to this topic.

1.3.1 Investigating User Habits

In order to be able to improve image/video browsing it is essential to collect and analyze data about how users interact with their smartphone and tablets. To gather this information, an extensive survey was performed. Participants were asked questions regarding their specific usage patterns when doing image or video related work with their devices. The gathered knowledge is reported as well as discussed in terms of implications.

1.3.2 Developing New Mobile Browsing Concepts

The core of this thesis is the introduction of several new interface concepts for mobile image and video browsing on smartphones and tablets. They utilize different approaches to improve the browsing experience of users. On one hand they use new strategies for content visualization and interaction. On the other hand they provide tools for filtering content in an interactive and iterative way. Some techniques that are used for those purposes include:

- 3D visualization
- Image sorting based on dominant color
- Sub-shot based video content segmentation

- User ergonomics
- Filtering dependent on various content properties (e.g. motion, keypoint density or color layout)

In total two tablet and four smartphone image browsing concepts and four tablet video browsing concepts are included.

1.3.3 Evaluation of Mobile Browsing Concepts

To be able to verify whether a browsing idea actually improves the state-of-the art it is necessary to test it. In comparison with other evaluation techniques, user studies are a method that is on one hand rather expensive and time consuming, but on the other hand can give great insights and feedback. This thesis reports about a number of such user studies. After an initial idea for a browsing concept, a prototype is built, which is then used by study participants to complete different search tasks. Their performance is recorded and afterwards analyzed. Three such reports are included in this thesis.

1.3.4 Measuring Performance for Content Analysis Purposes

In multimedia retrieval and browsing, images and videos have to be analyzed in order to gather more information about their contents. Typically, this process is linked with extensive processing needs. To deal with this problem on mobile devices, different approaches exist.

First, the processing can be offloaded to a server and performed beforehand (offline) or on demand via a network connection (online). The problem with this approach is that the content needs to be available in advance, or the device has to have a broadband internet connection at all times.

Second, the processing can be done directly on the device. This approach may require users to endure waiting times of different length, depending on the type of content and the chosen analysis method.

OpenCV[1] (Open Computer Vision) is an open source and freely available library of functions aimed at computer vision. It is widely used in the multimedia retrieval domain to perform different kinds of content analysis. Furthermore, it is available on a wide set of platforms, including Windows, Mac OS X, Linux, iOS and Android. Choosing the right method for the right environment can be a key to succeed. To be able to do this, measurements are required on the computational limits of different classes of devices. This fact is often overlooked and therefore to the best our knowledge no measurements in this domain exist for smartphone and tablets. Hence, in this thesis the results of two measurement setups are reported where commonly used OpenCV functions are evaluated regarding their performance on a diverse set of smartphones and tablets.

1.4 Structure

This thesis is organized in three big parts, in accordance with the discussion in the earlier contributions section. In chapter 2 the context of the thesis is presented. The research described in that work was performed to large parts in the Next-Generation Video Browsing project (NGVB). A short introduction and its goals are reported. Furthermore, the current state-of-the-art of research in the fields of image and video retrieval/browsing is described. Next, in chapter 3 the conditions and results of an extensive online survey are reported. In the following, the results are discussed and implications for new browsing designs are introduced. In chapters 4 and 5 the work and results of two user studies in the field of mobile image browsing are discussed. The first concentrates on image browsing on tablets and the second on image browsing on smartphones. In both cases novel interface concepts are introduced. Subsequently, in chapter 6 a novel video browsing concept, the Keyframe-Navigation-Tree Browser (KNT-Browser), is introduced. Moreover, it is evaluated in a user study of which the results are reported and discussed. After that, in chapter 7 the setup and report of OpenCV performance evaluations on mobile devices is given. Chapter 8 contains a short discussion about novel and published but not yet evaluated mobile video browsing interface concepts that incorporate the insights that were gathered in the earlier discussed research. Finally, the thesis is concluded in chapter 9.

[1] `http://www.opencv.org`

CHAPTER

2 Context

The work described in this thesis was performed for the most part in association with the Next-Generation Video Browsing Project (NGVB). This chapter includes an introduction of the project as well as an overview of important and recent work in the domain of (mobile) image and video browsing as well as retrieval.

2.1 The Next-Generation Video Browsing Project

The goal of the NGVB project (see Figure 2.1 for the projects' logo) is the investigation of novel video browsing opportunities by taking advantage of the capabilities of modern tablets and smartphones. Especially, novel browsing techniques should be discovered and evaluated. Furthermore, collaborative search utilizing mobile devices is another focus. The project was started in August 2012 and will be completed in February 2016. The project team consists of two project leaders, two full time members, and a few master students.

Figure 2.1: Logo of the Next-Generation Video Browsing project.

Two important sub-goals are (1) utilizing 3D visualizations for video browsing and (2) evaluating on-the-fly content analysis on smartphones and tablets. In earlier work it could be shown that a 3D visualization can in fact improve the browsing process compared to 2D scrollable grids [SAH14]. It is therefore the aim to further explore the possibilities that this visualization technique can offer in the given context. As the combination of content analysis and video browsing offers great potential, another aim is to test the capabilities of mobile devices to perform content analysis on-the-fly. This is important to investigate, as otherwise preprocessing or a server connection is mandatory. The project is structured into two primary work packages and two administrative work packages. In the following, the two primary work packages are described in greater detail.

Work package *one* (Novel Video Browsing Interfaces) focuses on designing, implementing and evaluating novel video browsing concepts for smartphones and tablets. Especially the utilization of 3D to enhance the browsing and navigation process is one of its main points. Furthermore, another important task is the investigation whether on-the-fly content analysis is possible with mobile devices and if yes, to what degree. Most of the work in this thesis was performed as part of this work package.

The *second* work package (Collaborative Video Browsing) investigated possibilities of solving video search and exploration tasks in a group of people. Each member of such a team would use a mobile device to participate. Especially in the professional domain such a system could be beneficial. For example, security camera footage could be much faster evaluated or it would improve the collaboration in the medical domain, as doctors could collaborate on recorded surgeries in a new way.

The project is currently in its final year and was successful in producing multiple research results that yielded in related publications.

2.2 State of Mobile Image Browsing

As mobile image browsing inevitably builds on the experiences and results of traditional image browsing/retrieval and content analysis, a couple of important works have to be mentioned before concentrating on the mobile domain.

Smeulders et al. [SWS+00] give an extensive overview about content-based image retrieval up to the year 2000. Rodden et al. provide further important insights regarding image browsing. On one hand, it seems that users prefer rather simple browsing solutions in contrast to very extensive but complex ones [RW03]. On the other hand, they showed first clues that arranging images similarity-wise could improve the browsing situation for users [RBSW01]. In the following, works of image browsing directly targeted at mobile devices like smartphones, tablets or PDAs are summarized.

Ganhör presents Athmos [Gan14], a context-driven visualization approach for image collections. The basic idea is a stripe of thumbnails where the center image (the focus) is displayed largest, while the rest of the thumbnails occupy the remaining space more or less distorted. Due to the nature of a rectangular screen this stripe is broken up into three stripes: a top stripe (starting range), a middle stripe (current range) and a bottom stripe (ending range). See Figure 2.2 for a screenshot of this arrangement. As users navigate through the collection with swipe gestures, the top and bottom stripes have to accommodate varying amounts of thumbnails, which results in different amounts of distortion.

With *MINI*, Gomi and Itoh [GI12] show a mobile visualization of multi-dimensional result datasets. Result items of the dataset (represented by images) are positioned in a 3D space. Three functions, which have to be defined by the user, are called to calculate the x-, y- and z-position of an item dependent on its attributes. Furthermore, their system lets users choose how to prioritize the result items (e.g., prioritizing certain keywords or attributes). It is possible to rotate and zoom this view with swipe gestures. The dataset is always displayed in such a way that high priority items are located in the top left corner.

Figure 2.2: Athmos: Focus- and context-driven mobile image browser by Ganhör [Gan14]

Schaefer et al. [STF+12] improve the traditional grid of thumbnails by using a hexagonal layout. Because of that, the images are visually more clearly separated, as can be seen in Figure 2.3. Moreover, they use image positioning and clustering based on color hue and intensity values. Similar images are close to each other while very similar images are grouped into one cluster represented by a special thumbnail. Users can explore the collection by zoom gestures and expand clusters by tapping on the respective thumbnails.

Khella and Bederson show an extension of their desktop image browser [Bed01] for mobile devices: [KB04]. It utilizes quantum strip treemaps to layout images and minimizes the amount of white space on the screen. Furthermore, it groups the images based on metadata or directory. With semantic zoom operations it is possible to focus on a certain group and use larger thumbnails.

Figure 2.3: Image browsing using a hexagonal layout by Schaefer et al. [STF+12]

An image browser utilizing creation dates and color similarity is presented by Kim et al. [KKC12]. They argue that images with similar creation timestamps most probably belong to the same *event* and should be part of the same cluster. Each of the clusters is visualized via a small collage in a grid-like interface. A grid of thumbnails then displays the contents of a cluster. Very similar images (duplicates, photos with only minor differences) are collapsed to a single entry, symbolized by a stack of photos.

Focusing on a problem that occurs when clustering based on creation date, Karlsson et al. [KJZ14] show a mobile photo browser with a multiscale timeline. Time-based clustering for events has the problem, that such events can last a few seconds (e.g., a group of people posing), a day (e.g., a visit of a theme park) or weeks (e.g., vacations). This makes it hard to apply time-based clustering as the scale always changes. Their browser therefore makes it possible to change clustering scales on-the-fly, dependent on their current needs. The individual clusters are displayed on a timeline that can be scrolled horizontally. A screenshot of the interface can be see in Figure 2.4.

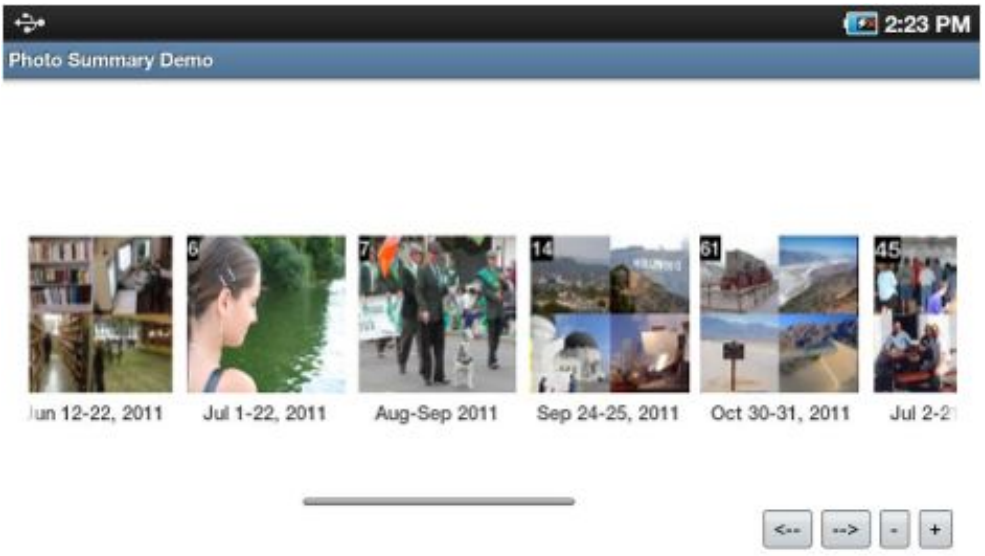

Figure 2.4: Image browser with a multiscale timeline by Karlsson et al. [KJZ14]

2.3 State of Mobile Video Browsing

As it is the case for mobile image browsing, also mobile video browsing builds on the results that were produced on traditional PCs before. An overview of the field is given by Schoeffmann et al. [SHM+10]. The ForkBrowser by de Rooij et al. [dRSW08] is a prominent example of combining a 3D visualization with similarity clustering.

In the following, research that is directly focused on mobile video browsing is reported.

iBingo [SFBJ08] is a system for collaborative search with mobile clients. A laptop computer initiates a search task by submitting a content-based query to a search engine. The shots in the result set of the query are then distributed among members of the search team. Each team member is equipped with an Apple iPod Touch device, which acts as mobile client. The team members decide on the relevance of their proposed results and give feedback to the central collaborative module. The module then reformulates and re-ranks the shots based on the given relevance feedback.

Miller et al. [MFF+09] present a mobile video browser for the Apple iPhone that focuses on live-events and lets users switch between multiple camera angles by using simple swipe gestures. They group their video application into four main context sensitive interfaces: a content browser, a video player, a multi-camera browser, and a history browser. Of special interest is the video player interface. During playback of a video, users can swipe for changing the camera. Swiping to the left switches to the camera located right of the current camera and vice versa. Furthermore, it can display annotations for the current video stream like names of players in a Hockey game. For this purpose users have to perform a single tap and select the appropriate annotation options. The information is then overlaid on the current video. Moreover, users can double-tap to bring up the multi-camera browser, which is a scrollable grid-view of all available camera-streams. Finally, their history browser provides functionality for going back to video sequences that were already selected earlier with a rich, video-edit-like interface.

Bursuc et al. [BZP10, BZP12] proposed *OVIDIUS* (On-line VIDeo Indexing Universal System). It is a distributed system that is structured in two parts: a search engine server that creates MPEG-7 descriptions of videos, and an interface server that enables video browsing via a web interface tailored to mobile devices. The UI enables a hierarchical browsing approach based on content segments. Navigation is performed vertically (scenes, shots, keyframes) as well as horizontally (segments of the same level).

Czepa et al. [CBH+12] show a streaming video player for smartphones, which incorporates users' attention. When users look away from their device during playback, the video is automatically paused. If they refocus their attention on the device the playback continues instantly. Their approach uses face detection in images recorded through the front camera of the smartphone.

Propane, presented by Ganhör et al. [Gan12], divides a landscape-oriented smartphone screen into four interaction areas: left, right, top and bottom. It can operate in three modes: *Standard Browsing*, *Advanced Browsing* and *Progressive Browsing*. When in *Standard Browsing* mode, tapping on the left or right areas once navigates the video to the previous or the

next frame. When a finger is held on the areas the video is fast forwarded/rewinded. The seeking speed is dependent on the vertical position of the finger in the area. The *Advanced Browsing* mode further enables users to slow down a video for slow-motion playback using the same areas. Last, the *Progressive Browsing* mode is a combination of both earlier mentioned modes.

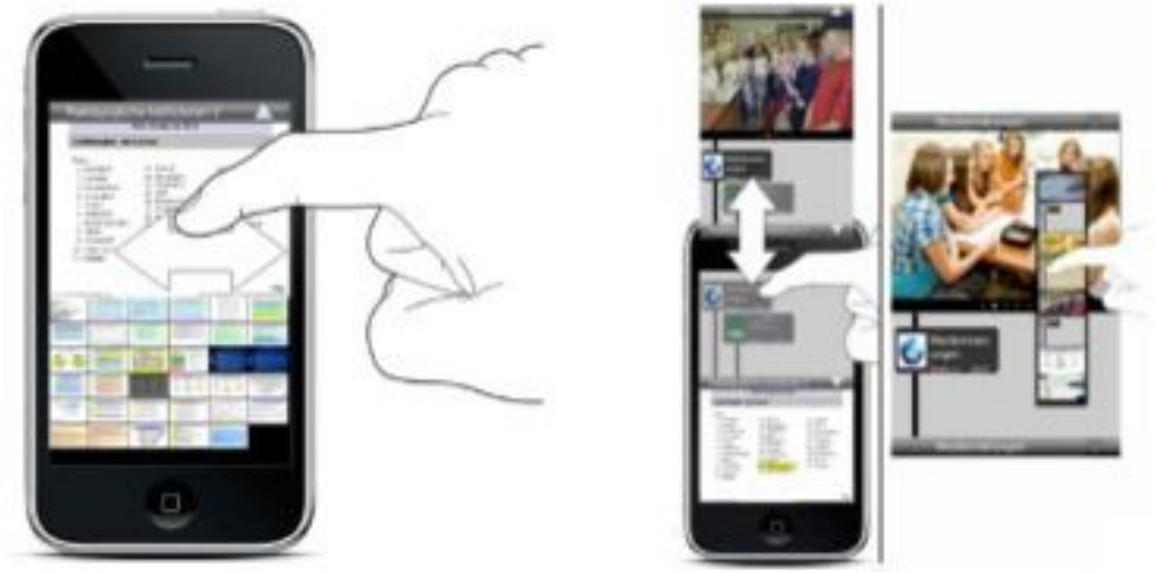

Figure 2.5: Wipe'n'Watch: navigation in eLecture-videos by Huber et al. [HSL+10]

Huber et al. [HSL+10] propose a mobile video browsing approach tailored to e-lectures: *Wipe'n'Watch*. Central to their idea are horizontal and vertical wipe gestures. They enable users to navigate between the keyframes (i.e., e-lecture slides) of a video and between topically overlapping videos (e.g., keyframes of two videos cover the same topic). The proposed visual interface is divided horizontally, as can be seen in Figure 2.5. At the top the content of the video is shown. At the bottom a grid of all keyframes of the current video is displayed. The presence of an topical overlap is indicated by a small arrow in the top right corner. Horizontal wipes navigate between keyframes. A vertical wipe during an overlap first shows a list of related keyframes of other videos. When users tap on one of them the interface is further scrolled down and displays the interlinked keyframes of the chosen video.

This process can be repeated as long as there are further semantically overlapping videos. Moreover, a browsing history can be used for direct access to any of the earlier viewed videos.

One handed use is important for smartphones. Huerst and Merkle [HM08b] consider this when they are presenting a mobile video browser for one-handed operation. The device is held in landscape orientation while users scroll through a list of keyframes. Interaction is performed only with the right thumb on the right side of the touchscreen. It provides automatic scrolling of a list of keyframes, which can be manipulated by thumb gestures. The list can also be scrolled manually. The video position can be set by positioning a keyframe in the middle of the list and tapping the screen once.

A very important work for the space of mobile video navigation is the *mobile zoom-slider* presented by Hürst et al. [HGW07b] in 2007. It enhances the functionality of seeker bars by allowing to seek with different granularity levels. Depending on the vertical position of a stylus touching the screen, a different navigation resolution is used, as can be seen in Figure 2.6. At the top of the screen a fine granularity is used with minor temporal jumps. When operated at the bottom of the screen, it uses a coarse granularity with major temporal jumps. Various video players on smartphones and tablets later adopted this idea.

Buchinger et al. [BHH+10] show a video player interface for a smartphone or a similar mobile device that is controlled by physical and touch interaction. To pause a video users can position the device face down on a table or upside down in a pocket. Playback resumes as soon as the device is positioned into landscape orientation. Switching between channels is realized by swinging the device up or down. If users want to choose channels they have to orient the device into portrait orientation. Furthermore, the system allows to control the audio volume by swiping over the screen from left to right (increase) or from right to left (decrease). A diagonal swipe from top left to bottom right mutes the sound completely, whereas a swipe in the opposite direction restores the original volume. Fast forwarding and rewinding can be activated with half-circular swipe gestures, either to the right (fast forward) or to the left (fast rewind). Finally, a diagonal swipe from top right to bottom left of the screen positions the video to the beginning.

Figure 2.6: A video timeline with different granularity levels by Hürst et al. [HGW07b]

Meixner et al. present the concept of a mobile video browser that incorporates annotations to give users a non-linear video experience [MKK11]. Their SIVID player presents users a split-interface. One side is used for the video playback, while the other side is used to display annotations. Such annotations can be simple texts, images, or links to other positions in the video. Furthermore, if multiple annotations are available for the current position in the video an annotation stack is displayed. Users can tap on the elements on the stack to get to the corresponding annotation, or display them in fullscreen mode.

Lux and Riegler [LR13] propose a mobile video player that allows to annotate video by speech, speech-to-text, and in-frame sketches as well as bookmarking of frames and segments. The annotations are indicated on the time slider and automatically displayed on playback of the corresponding time positions.

Various interfaces and interaction models for mobile video browsing have been proposed by Hürst and Meier [HM08a]: *Dynamic flicking* enables users to quickly scroll through a video by dragging a stylus across the screen. The direction determines whether to seek forward or backward. The drag distance determines the seeking speed. *Elastic panning* uses an elastic slider metaphor to enable forward or backward seeking in a video. *Constant panning* is similar, but the thumb is fixed and scrolling speed remains the same if the stylus is not moved. In *dynamic panning* the initial touch position of a stylus is set equal with the current position in a video. The left and right edges of the screen then represent the beginning and the end of the video, while dragging the stylus in either direction navigates the video. *Constant flicking* enables controlling an automatic, constant scrolling speed and direction with flicking on the screen. In contrast, *dynamic flicking* decreases scrolling speed automatically over time.

Figure 2.7: HiStory: video browsing with a hierarchical storyboard by Hürst and Darzentas [HD12]

Utilizing a storyboard-like arrangement, Hürst and Darzentas [HD12] show a hierarchical approach for mobile video browsing. A screenshot of the default view can be seen in

Figure 2.7. A grid of video keyframes are displayed on the screen. They are uniformly sampled and represent chronologically ordered video sequences of the same length. A tap on one of the keyframes refills the grid with keyframes of corresponding sub-sequences. This can be continued until each keyframe represents a single frame. A narrow sidebar on the right side tells users their current hierarchy level.

Zhang et al. present a mobile video system for collaborative annotation of videos as well as video navigation with sketch gestures [ZML+13]. They describe a scenario where two users are browsing the same video. Sketches, which are entered through the touchscreen, are immediately visible to the other user if they are in the same part of the video. Users can also share their current position and annotations. Such shared information is indicated with a small image presented at the bottom right of the screen. Users can communicate with each other via a built-in chat function and navigate in the video by applying predefined touch gestures. For example, dragging both fingers from the sides to the center of the screen opens the playlist, drawing a line from right to left starts fast reverse, and a single tap on the screen starts playback or pauses the video.

A collaborative video browsing tool has been proposed by Cobarzan et al. [CHDF14]. The system is composed of multiple mobile clients implemented on Apple iPad devices and a single server that manages communication between the clients as well as content-based search requests. Users are able to search and browse a single video as well as video collections with their tablets. The interface presents uniformly sampled keyframes with a storyboard-like arrangement of thumbnails and allows users to perform content-based queries (e.g., color-based filtering). Such queries are sent to the server, which returns matching sequences. These sequences are then visualized through a 3D ring of keyframes [SA12a] and can be further inspected by a simple selection (i.e., a tap gesture). The tool also shows which parts of the video have been already viewed, either by the current user or other search team members. Also, the content-based queries as well as set bookmarks of all team members are shared among the group.

Similar to the approaches of Huber et al. [HSL+10] and Hürst et al. [HGW07a], Schoeffmann et al. [SCB14] propose another wipe-based video navigator. Horizontal wipes on the touchscreen first pause the playback of a fullscreen video. Then, the current frame is *moved* outside of the screen, according to the direction of the wipe gesture. At the same time an earlier or later frame is moved into the screen. This behavior is meant to mimic browsing a photo collection. When the finger is lifted the playback continues from the new position. It also adopts the idea of different granularities of the ZoomSlider [HGW07b]: when the wipe is performed at the top of the screen, larger time skips are applied, while shorter time skips are used when the wipe is performed at the bottom of the screen.

Hürst and Hoet [HH15] perform a comparison between two different approaches for mobile video browsing on tablets. They compare a storyboard design with a slider-based approach. The storyboard design features a scrollable grid of 5x5 keyframes. Furthermore, their slider-based approach uses a vertical seeker bar on the left and a vertical filmstrip for fine granular navigation on the right side of the screen. In their study they could confirm the usefulness of both designs but no approach showed significant better results over the other. Therefore, they suggest to integrate both approaches into a single interface.

CHAPTER

3 Mobile Media Collections: A Survey

In order to investigate better alternatives to default image and video browsers, it is necessary to get a better understanding of the current status quo. How many images do users store on their devices? Where do they originate from? How often do they capture new images? Are they willing manage and describe their images on their own? These questions have great impact on how an improved solution has to look like.

Figure 3.1: Motorola Razr V3 [Jur04]

Kindberg et al. [KSFS05] performed a study on digital image capturing behavior in 2005. However, the problem is that smartphones were not widely adopted yet at that time. In their study they investigated feature phones with built-in cameras instead. Feature phones have very limited functionality compared to smartphones. They can be used for voice calling, texting and sometimes very basic internet and multimedia purposes (like storing a couple of photos in low resolutions). For an example see the Motorola Razr V3, a

popular phone at that time, in Figure 3.1. They report that the average amount of images on the evaluated devices was around 44 images.

Another more recent study performed by Thakur et al. [TGE11] focused on work-related scenarios only. About 90% of their participants stated that they would use their phone at least once a week to take photos. They state that 45% had more than 100 photos stored on their smartphones with an additional 3% having more than 1000 images stored on their device.

Puikkonen et al. presented a study of video creation with mobile phones [PHBM09] where participants reported that they use their mobile phones for video creation only once or twice per month.

Furthermore, two surveys that were performed by Suite48Analytics were discovered, which are unfortunately not freely available. Nevertheless, in their abstracts they state that among 1004 interviewed users, more than 76% store more than 25 photos on multiple device types (smartphones, computers, etc.) [Sui14a]. Moreover, they show that among 1000 interviewed consumers in North America 58% of the participants, who have taken at least ten photos in the last three months, exclusively used their smartphone to capture them [Sui14b]. They also note that 33% of their respondents use a combination of smartphone and digital camera and that 5% in addition regularly take photos with their tablets.

Since none of the available surveys were entirely satisfactory, a custom survey was designed, which will be discussed in this chapter[1]. An important objective of this study was to structure the study in a way that provides valuable insights about usage patterns of mobile image/video capturing, which can be used as input by the design and research community ([SHM+10], [BCD+12]) interested in improving interfaces for photo and video management.

[1]Please note that the contents of this chapter are adapted from [HSAL15]

3.1 Study Design and Participant Statistics

The study was designed as an online web-based survey in the German-speaking region (mostly Austria and Germany). To spread the word about it, social media (Facebook, Google+, Twitter) and mailing lists were employed. The survey included 41 questions that were grouped in three sections: a smartphone section, an optional tablet section and finally, a demographic section.

In the smartphone and tablet related sections the participant were asked various questions about their mobile image and video collections:

- Period of ownership
- Storage capabilities
- Manufacturer and OS
- Number of images and videos stored locally (no cloud or similar services)
- Origins of the stored images (captured with device, downloaded, etc.)
- Organizing strategy (if any)
- Backup strategy (if any)
- Motives and intents

The questions of the first two sections (smartphones and tablets) were the same except for slight formulation changes due to the nature of the devices. Furthermore, when participants completed the smartphone section they were asked whether they also were in possession of a tablet. If participants negated that question the entire tablet section was skipped. In either case, the last (demographic) section asked about gender and age of the participants, which completed the survey.

Over a timespan of three weeks a total of 215 participants completed the survey. Of these, 125 were female and 90 were male. Moreover, of these 215 participants in total 82

also were in possession of a tablet. Of these 82 tablet users, 38 were female and 44 were male. The average age was around 29.9 years (s.d. = 9.0), with the male average age at 31.6 years (s.d. = 9.8) and female average age at 28.7 years (s.d. = 8.2). When the data is restricted to participants who also own a tablet, the average age was around 31.6 years (s.d. = 10.3) overall. Furthermore, the average age of female tablet users was at 30.2 years (s.d. = 9.6) and male average age was at 32.8 years (s.d. = 10.9).

3.2 Results

The results of each of the before mentioned categories will be reported in their own sub-chapter. Each report includes the figures of both, smartphones and tablets. Implications and possible explanations are also included.

3.2.1 Ownership

The aim of this question is to investigate how familiar the participants to their devices were. Most of them used them for at least a year or more. This is important, since their media collections are the focus. Collections typically grow over time so it is necessary to have participants that owned their device long enough. Otherwise, the sample would not be representative.

In case of smartphones the data indicates that most participants already had them for more than two years (70.2%) or at least one year (20.0%). Other participants had their smartphones between six months and one year (6%) or for less than six months (3.7%).

Many participants indicated a shorter time regarding their tablets (more than one year: 34.1%, more than two years: 37.8%). Additionally, 15.9% indicated that they had their tablet between six months and one year and 12.2% reported that they had it for less than six months. The numbers are visualized graphically in Figure 3.2.

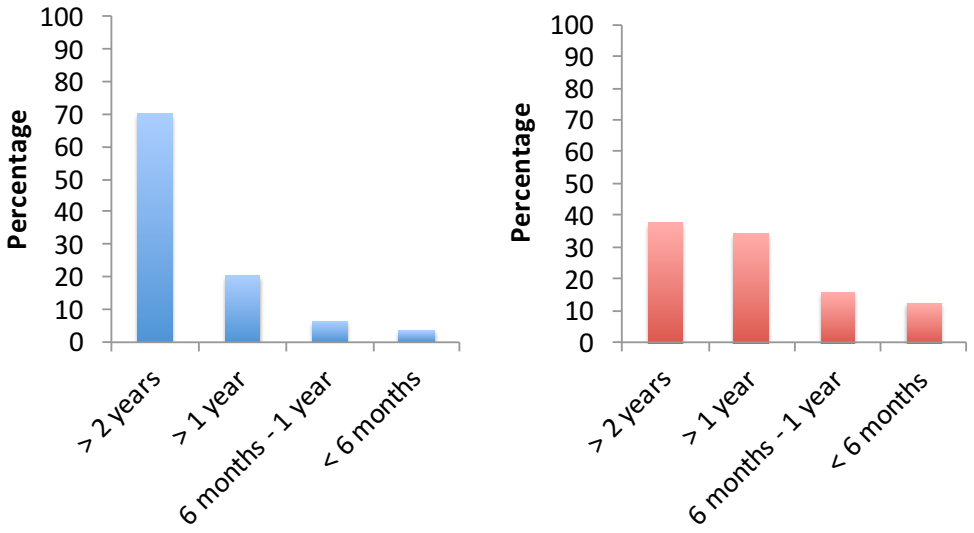

Figure 3.2: Length of ownership for smartphones (left) and tablets (right).

3.2.2 Storage Space

Participants had to indicate how much storage space their devices feature. In case of smartphones, 20% had 8 GB, 39.5% 16 GB, 14.4% 32 GB and 2.3% had 64 GB of storage space (see Figure 3.3). Moreover, 18.6% of the participants could not tell how much storage their device offered. Participants who entered other storage space sizes were around 5.1%.

In case of tablets, 7.3% of the participants indicated that their devices offered 8 GB, 28% 16 GB, 26.8% 32 GB, 15.9% 64 GB, 4.9% 128 GB and 1.2% reported 256 GB. Furthermore, 15.9% of the participants reported that they were not aware of how much storage space their tablet offers (as can be seen in Figure 3.4).

It was expected that 16 GB was the most used option for smartphones. Most phones today are produced and sold with this memory size and for many people it seems to be the best option in terms of space and cost. Interestingly, people seem to prefer to have a little bit more space available when it comes to tablets, as the 16 GB and 32 GB options are head to head. It will be interesting if this has any indication whether our participants actually store more media on their tablets as a result.

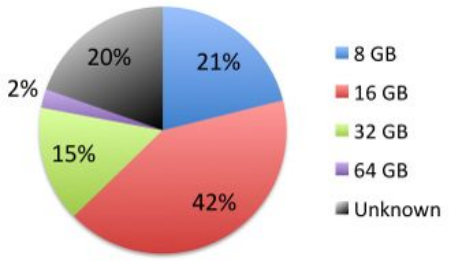

Figure 3.3: Distribution of Smartphone storage capabilities.

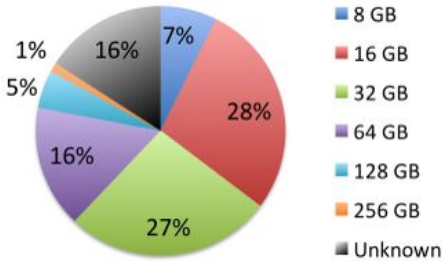

Figure 3.4: Distribution of Tablet storage capabilities.

Surprisingly, also the amount of participants who had no idea about their device's storage capabilities was quite high. It seems that for a notable amount of people storage space does have no influence on their buying decision.

3.2.3 Manufacturers and Operation Systems

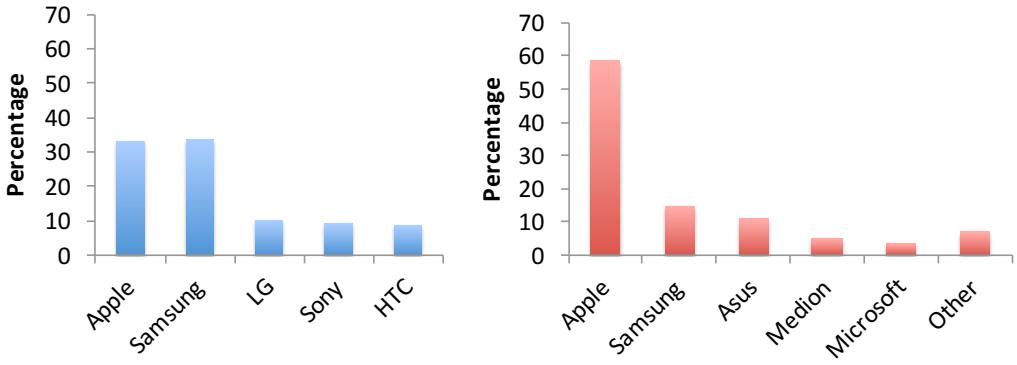

Figure 3.5: Distribution of smartphone (left) and tablet (right) vendors.

Most smartphones of participants were produced by either Samsung (33.5%) or Apple (33.0%), followed by LG (9.8%), Sony (9.3%) and HTC (8.4%), as can be seen in Figure 3.5.

Other manufactures amounted together for 6.2%. The most reported smartphone OS was Android with 62.8%, followed by iOS with 32.1%, Windows Phone with 2.3% and Blackberry OS with 0.5% (see Figure 3.6). Furthermore, 2.3% of the participants noted that they were not aware what operating system they had installed on their device.

Compared to worldwide smartphone vendor figures of IDC for the third quarter of 2014 [IDC14b], the participant sample is a little bit off. The numbers agree on Samsung and Apple as the leader (although our percentages are higher). Xiaomi and Lenovo as Vendors are not existent in our sample. This is probably a result of the strong focus of those companies on the Chinese market and our focus on the German-speaking region. Interestingly, IDC reports a huge amount of *other* manufacturers (49.3%) to which no further information is given.

When comparing the market shares of smartphone operating systems to IDC [IDC14a], our sample corresponds nicely with the order but shows some deviations in terms of percentages. IDC reports larger differences between Android and iOS (84.4% and 11.7%), while the other numbers are much more close (deviations of only 0.6 percentage points at max).

Most tablets were manufactured by Apple (58.5%), followed by Samsung (14.6%), Asus (11.0%), Medion (4.9%) and Microsoft (3.7%). The percentages of other manufacturers of which each one was below 1.5% can be summed to 7.2%. Dominant operating system on tablets was iOS with 51.2%, followed by Android with 30.5% and Windows with 8.5%. Another 9.8% noted that they were not aware of the name of their tablet operating system (see Figure 3.7).

A comparison with IDC numbers of the third quarter of 2014 [Sta14] shows that top three vendors are the same for our sample (Apple, Samsung, Asus), although Apple shows worldwide a strong difference with only 22.8%. It has to be noted that Medion is a vendor that is strongly focused on the German-speaking market and therefore is absent in worldwide numbers. Also Microsoft is better represented in our sample than internationally. Moreover, IDC marks a huge amount of 41.8% as *other* vendors.

Surprisingly, in terms of tablet OS percentages our sample deviates strongly with international numbers of IDC for 2014 [IDC14c]. They see Android at the top with 67.7% while iOS ranking second with 27.5%. The reason for this very likely is the already stated high number of *other* vendors in their report. It can be expected that most of them are rather cheap Android tablets, sold in not so developed markets.

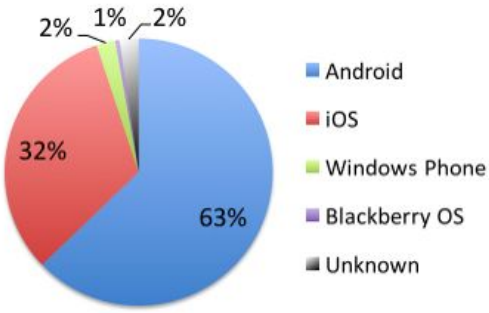

Figure 3.6: Distribution of smartphone operating systems.

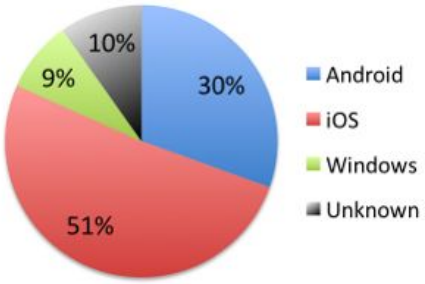

Figure 3.7: Distribution of tablet operating systems.

3.2.4 Stored Photos and Videos

Participants were also asked to tell how many photos and videos they had stored on their devices. In case of smartphones, an average image count of **440.3** (s.d. = 503.3) was discovered, with a minimum value of 0 and a maximum value of 2166.0. The average was calculated after removing outliers with Tukeys outlier labeling method [Tuk77] and an adjusted multiplier of 2.2 [HIT86]. Furthermore, an average video count of **10.6** (s.d. = 12.4) with a minimum value of 0.0 and a maximum value of 58.0 was calculated. For tablets an average image count of **113.5** (s.d. = 189.6) could be calculated, with a minimum value of 0.0 and a maximum value of 826.0. Interestingly, the difference to smartphones is quite remarkable. The same can be seen when focusing on videos. An average video count of **4.6** videos (s.d. = 6.9) was calculated, with a minimum value of 0.0 and a maximum value of 25.0.

This question was especially important, as it provided a first direction for how many images and videos we should expect when developing new browsing solutions (for general purpose). The effects can be seen later in the chapters 4 and 5, where four interfaces will be presented (including different versions for smartphones and tablets) that operate with collections of up to 400 images.

Moreover, the video counts were in both cases rather low. As a consequence, focus of this thesis will be to improve video browsing inside a single video insteak of large collections. In chapter 6 a new video browser for such purposes will be introduced and evaluated.

Also, a contradiction becomes visible when comparing these numbers with storage space choices. As was discovered, people choose higher storage options for tablets than for smartphones. Surprisingly, they store in fact less media on their tablets. In lack of further data, the following two theories could be the cause for this behavior.

On one hand, it could be caused by mental reasons. Users expect to have more data on their tablets in the future, since it is a larger device. This expectation turns out to be false over time, as they use it in a different way than what they originally planned.

On the other hand, tablets are not only used for images and videos. The additional storage needs may originate from other apps that are installed, like navigation software, games or music. However, no additional information that could further explain this circumstance was collected in this study.

3.2.5 Media Sources

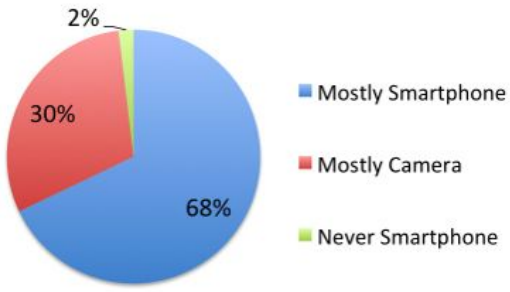

Figure 3.8: Smartphone Photo-Capture Habits

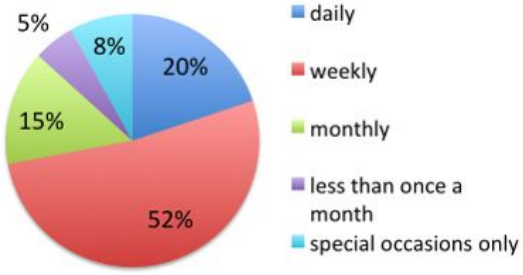

Figure 3.9: Smartphone Photo-Capture Frequency

In the survey participants were also asked if they use their devices as the primary tool for taking photos and in which intervals they add new photos. In case of smartphones 67.9% of the participants noted that they primarily shoot new photos with the smartphone (see Figure 3.8). Another 30.2% indicated that they preferred in most cases a normal camera for photo shooting. The amount of people that never use their smartphone for taking a photo was relatively small with 1.9%. Additionally, the time interval for new photos created with their smartphones was rather short, as can be seen in Figure 3.9. Most participants noted that they would take new photos on a weekly (52.1%) or daily (19.9%) basis.

Participants were also asked from where most of the stored images on their smartphone originated from (Figure 3.10). The majority (88.6%) reported that most of them were shot directly with their devices. The second option (from instant messaging services) was far behind with only 6.2%. Other options like synching photos from a PC, a camera, a USB-stick, from the internet or e-mails or from cloud services were all individually below 2%.

The numbers confirm our initial expectation that users are replacing their digital point-and-shoot cameras with their smartphones. Furthermore, it appears that they use them on a rather short recurring cycle, as more than two thirds take new photos daily or weekly. The need to focus on smartphones therefore definitively exists. As a result, over 80% of the images on smartphones are photos that are shot directly with the device.

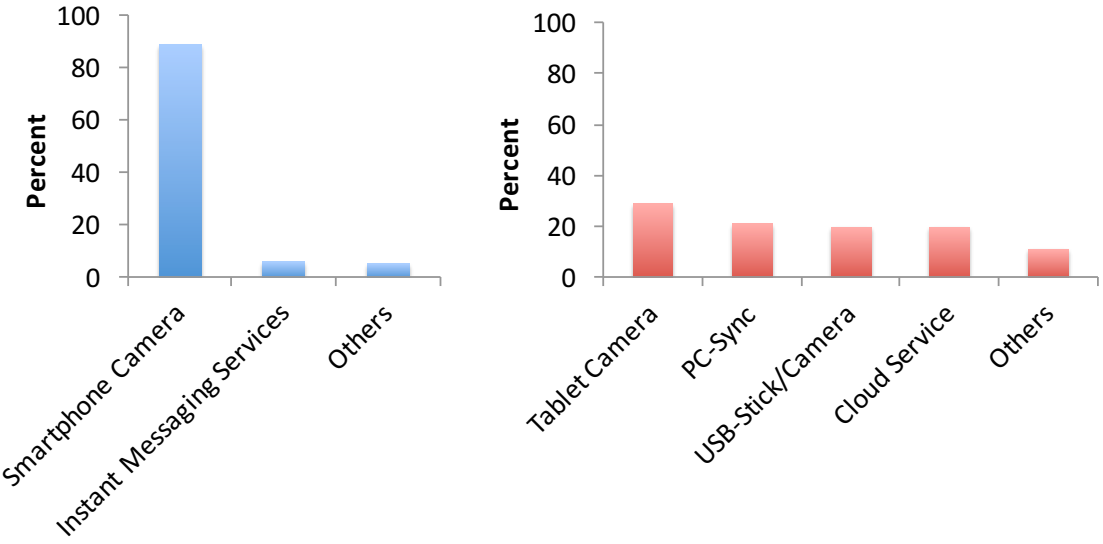

Figure 3.10: Image origins on smartphones (left) and tablets (right).

The numbers were completely different for tablets. Only 2.4% use their tablet as primary device for photo shooting. Photos are shot in most cases with their smartphones (40.2%) or with a dedicated camera (14.6%), as can be seen in Figure 3.11. Also, 42.7% indicate that they never shoot any photos with their tablets. In terms of time intervals participants indicated that they would use their tablets less than once a month for taking a photo (66.0%, see Figure 3.12).

Tablet users were asked the same question about file origins. The answers were much more diverse (Figure 3.10). The four dominant origins of images on tablets are: (1) directly shot with the tablet (29.0%), (2) synced from a PC (21.0%), (3) copied from USB-stick or camera (19.4%), and (4) synced from a cloud service (19.4%).

The tablet related numbers indicate that users avoid using them as cameras. Nevertheless, tablets seem to be used as viewing and browsing devices as the various kinds of image sources show. Especially interesting is the high amount of cloud services and their

absence with smartphones. This could be explained by the context in which tablets are used. Typically, they are used in-house where Wi-Fi is available. As a result, bandwidth is no restrictive element anymore and cloud services become more attractive. Smartphones on the other hand have to operate in more restrictive environments.

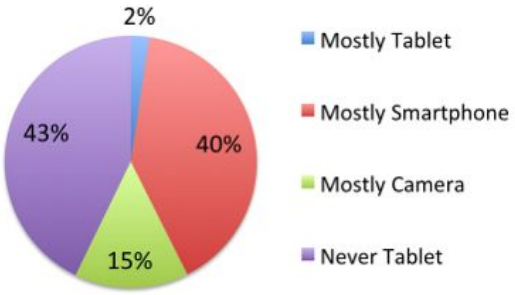

Figure 3.11: Photo origins on tablets

Figure 3.12: Tablet capture frequency

3.2.6 Image Organization and Photo App Usage

Participants were also asked if they would organize their images in manually created folders or albums. It seems that in case of smartphones manually organizing photos is quite unpopular, with 56.2% noting that they would never do something like that. Another 22.4% noted that they do it rarely, 14.3% would do it for certain, special images, and only 6.7% do it on a regular basis. In case of tablets a quite similar pattern of answers was received. Most participants do not organize their images on their tablets (56.5%) or only in certain cases (21.0%). Furthermore, 12.9% noted they would do it rarely and 9.7% said they would always organize their images in such a way on their tablet.

Another question was if they would use any special photo apps to organize their photos, which does not seem to be the case on smartphones, with 94.8% choosing *No* as answer. The same is true in case of tablets with 96.8% not using any kind of such apps.

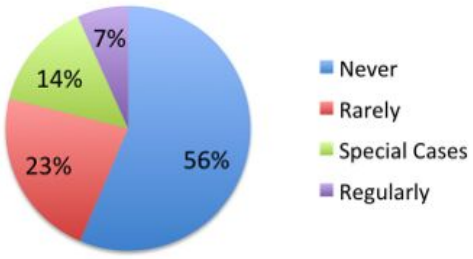

Figure 3.13: Smartphone organization habbits

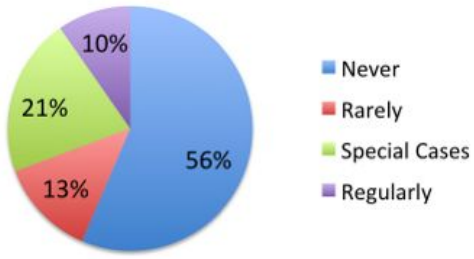

Figure 3.14: Tablet organization habbits

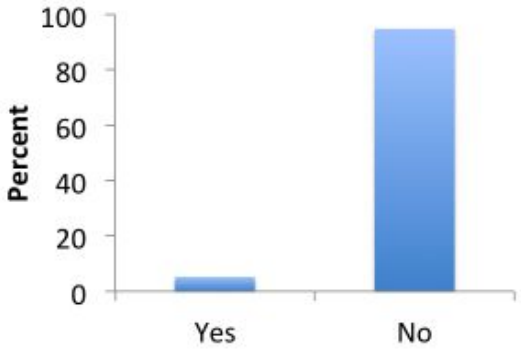

Figure 3.15: Usage of specialized photo apps on smartphones

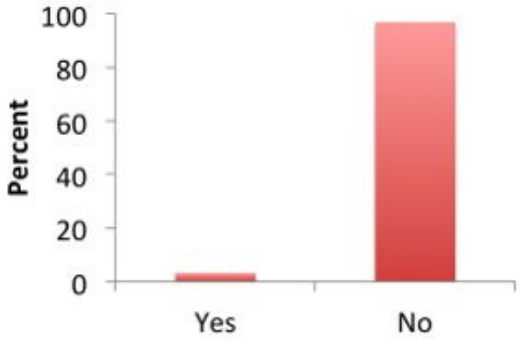

Figure 3.16: Usage of specialized photo apps on tablets

What can be concluded from these numbers is, that the majority of users do not organize their images in any way and also do not use any special management apps. This can be interpreted in two ways: (1) there is no need for any further investment in image browsing technologies because users do not care or (2) current solutions are too complicated and inflexible to use, and users are still looking for a better way. The work in this thesis is following prediction two.

3.2.7 Backup Interval

The last question was about participant's backup strategy. They could choose between the options *automatic (i.e., cloud-based)*, *daily*, *weekly*, *monthly*, *less than once a month* and *never*. When asked in case of smartphones most participants marked the options *less than once a month* (34.0%), *automatic* (24.1%), *never* (23.6%), *monthly* (11.3%), *weekly* (4.7%) and *daily* (2.4%) as can be seen in Figure 3.17. In case of tablets the most submitted option was *automatic* (35.5%) followed by *never* (27.4%), *less than once a month* (21.0%), *monthly* (8.1%), *weekly* (6.5%) and *daily* (1.6%). The distribution can also be seen in Figure 3.18. l

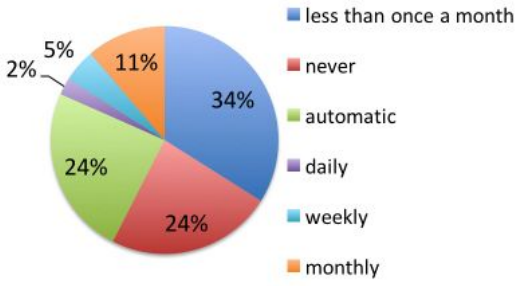

Figure 3.17: Backup strategies on smartphones

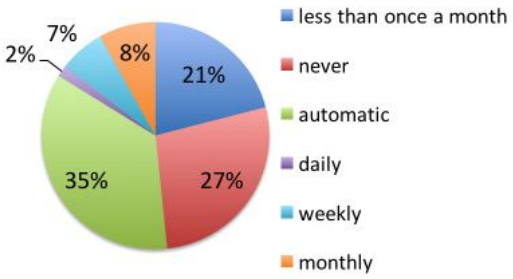

Figure 3.18: Backup strategies on tablets

The figures show clearly that backup strategies are more an afterthought for users. Either, they do not bother at all or only in very limited fashion. What is also visible is that automatic backup solutions are on the rise. They are especially popular on tablet devices, which corresponds with the earlier described usage context. For automatic solutions to work properly a reliable connection is required, which is more often the case for tablets than for smartphones.

3.2.8 Motives and Intents

Besides technical and demographic parameters also popular motives and capture-intents of study participants were investigated. Ten different options for popular motives were available, including *people*, *holiday pictures*, *events*, *landscapes*, *animals and pets*, *buildings and*

architecture, *shopping and product photos*, *food and eating*, and *flowers*. An additional open question allowed for entering motives that were not covered by the ten categories. Answers were presented in an ordinal scale, i.e. *never*, *rarely*, *sometimes*, *often*, and *very often*. Most popular motive was *people* with an average of **2.69** in between *sometimes* (2) and *often* (3).

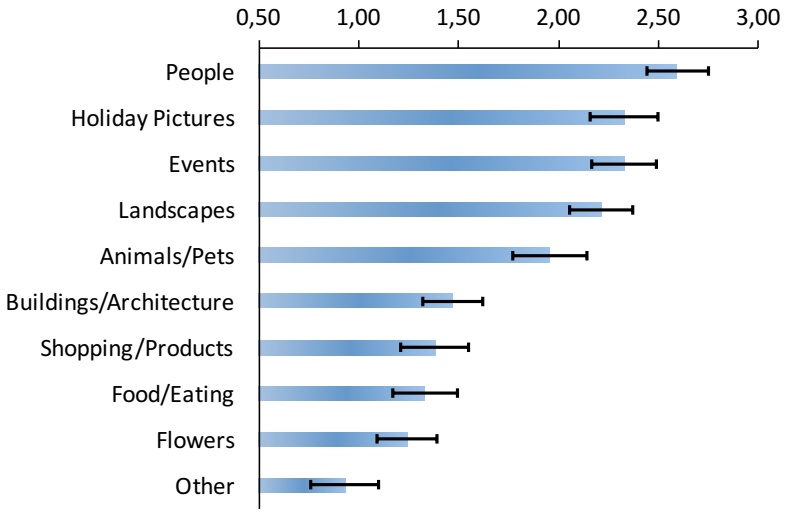

Figure 3.19: Motives of users sorted in descending popularity (0..never, 1..seldom, 2..sometimes, 3..often, 4..very often).

However, based on the confidence intervals for the average value it can be seen that the motives *people*, *holiday pictures*, *events* and *landscapes* are generally more popular than the others. On the other hand, it seems that *buildings*, *shopping*, *food* and *flowers* are noticeable less popular to each of the four above mentioned as the confidence intervals do not overlap (see Figure 3.19). *Other* is actually below 1 in average, meaning that it ranges between never (0) and rarely (1).

While additional motives were not as popular as the main categories, there was an interesting additional category mentioned: 28 out of 199 participants noted that they were taking photos of *documents, notes, blackboards and flip charts*, to capture and store text or graphics of documents. Eight more participants noted *funny, humorous and strange* (in an entertaining sense) motives, while seven participants noted *cars* as a popular motive.

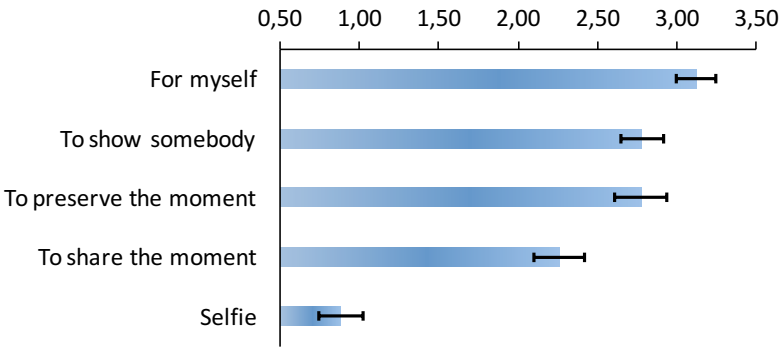

Figure 3.20: Intents of users sorted in descending popularity (0..never, 1..seldom, 2..sometimes, 3..often, 4..very often).

Besides the popular motives survey participants were asked about their intent or the intended use of the captured pictures. Most popular intent was *for myself* with an average of **3.12** in between *often* (3) and *very often* (4). Based on the confidence intervals, it seems to be more popular than *to show somebody* (e.g., showing off) and *to preserve the moment* (see Figure 3.20). This is a non-obvious results and leads to the conclusion that in the age of social media still a significant part of the images taken on a mobile phone are intended for personal use only. Other intents that involve sharing like *to share the moment* (avg. 2.26) and *selfie* (avg. 0.88) are also less popular intents. When examining additional intents not covered by the main categories, *to document something* was discovered as most popular. This goes along with the earlier findings on the motives, additional to the ones given in the questionnaire, where taking photos of documents was often mentioned by participants.

3.3 Discussion

In this discussion two significant differences are reported that were discovered during analysis of the data. Both concern the amount of stored images on smartphones: a difference in gender and a difference between manufacturers and operating systems. Furthermore, correlation hypotheses are reported that were tested in the gathered data.

3.3.1 Gender Differences

In the data analysis a significant difference for stored images between women and men was discovered. The data did not show a normal distribution as a visual inspection with a Q-Q-Diagram showed. Therefore, the data was analyzed with the Mann-Whitney U test, as it does not require a normal distributed dataset and is the non-parametric alternative to the otherwise typical independent t-test. However, it has the requirement that the distribution of both groups is similar in shape, which was the case, as a visual inspection revealed. The median count of photos was statistically significantly higher for females (384.0) than for males (200.0) as can be seen in Figure 3.3.1, U = 4217.0, z = -2.650, p = 0.008. The same test did not reveal a significant difference when repeated for tablets with the same preconditions.

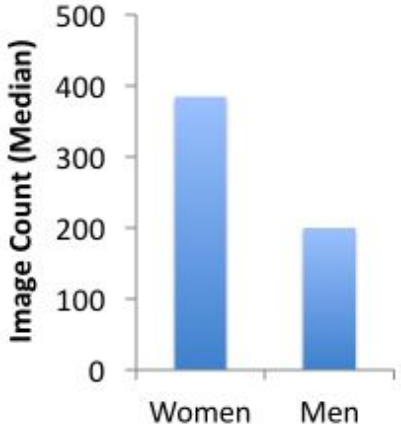

Figure 3.21: Images-Counts (medians) between women and men

Speculating about the reasons for this result is difficult, but it seems that female participants make more photos with their smartphones in general. It would have been interesting to know, whether the captured images could be assigned a special group of photos (like shopping, portraits, group photos, etc.). Unfortunately, this data was not available. It could be that female users are more likely to create images of memorable events or communicate more via images (by capturing them first and then sending them with e-mail or instant messaging services). As noted, a follow-up survey would be required to answer this question properly.

3.3.2 Differences between Operating Systems

Another significant difference that was discovered concerns operating systems related to the image count. This time, the differences exist for both device types.

In the smartphone analysis *Windows Phone* (five participants) and *Blackberry OS* were excluded since their results would not be representative enough given the low number of participants (five and one participants). Therefore, the focus was on a comparison between *Android* and *iOS*. Similarly to the analysis between genders, the data did not show a normal distribution for which reason the Mann-Whitney U test for statistical evaluation was chosen. The median photo count was statistically significantly higher for iOS (500.0) than for Android (200.0), $U = 2913.0$, $z = -3.932$, $p < 0.001$. In case of tablets, the Mann-Whitney U test showed that the median photo count was again statistically significantly higher for iOS (286.0) than for Android (30.0), $U = 188.0$, $z = -2.561$, $p = 0.01$. Figure 3.22 visualizes these differences visually.

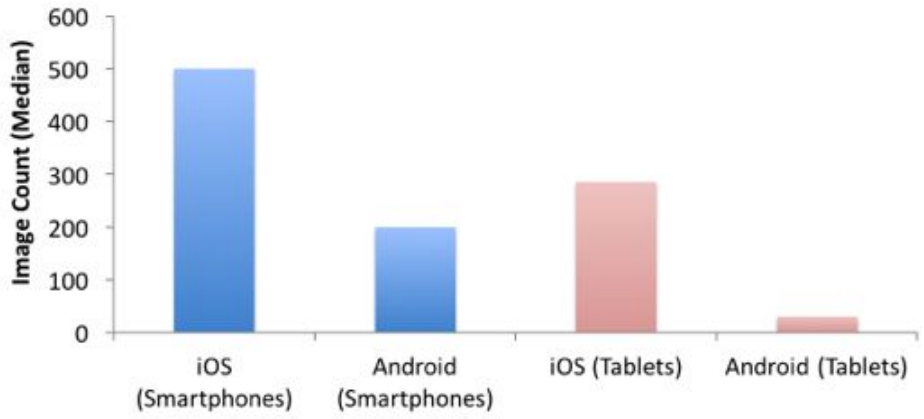

Figure 3.22: Different amounts of images (medians) between the two biggest operating systems

The result could be explained when considering the different camera qualities that are used in both cases. iOS only runs on iPhones, which in general have high praised photo quality. In contrast, Android runs on a wide variety of devices with different specifications

and price-points. Especially cheaper Android smartphones do not include camera modules of great quality. Sometimes, also pre-installed photo software can be slow and complicates the capture process. Moreover, Apple products are supported by extensive marketing efforts that also concentrate especially on the photo capturing capabilities of the devices. Users could be more conscious about photos shooting because of these efforts and therefore make photos more often.

3.3.3 Correlation Hypotheses

The gathered data was investigated for different kinds of relations between various factors. Although it was not possible to detect any meaningful associations the lack of those could still be of interest for researchers in the field. To investigate possible relations first a visual inspection on the appropriate data for monotonic increasing or decreasing graphs was performed, as they are a preference for performing a Spearman's Correlation test. When this visual inspection failed the hypothesis of a relation between the factors was dismissed.

What was discovered first is, that there seems to be **no relation between stored photos and storage sizes of the devices**. One could argue that the more storage is available participants could be more likely to store more images. However, no indication for such an association could be found in the data.

It was also analyzed if there exists a relation between the amount of stored photos and the backup interval. There seems, however, to be **no relation between how many images people have on their devices and how often they would backup their device**.

Another expectation was, that participants who had many images on their devices would be more likely to organize them in specific folders or albums. The gathered data indicates nothing in that direction. There seems to be **no relation between organizational habits and how many images are on the device**.

Furthermore, a possible relation between the interval in which users take new photos and the amount of images on the devices was investigated. One could argue that a short interval

should indicate a larger amount of images on the device. Nevertheless, in the data no indication that supports this claim could be found. **No association between the recording interval of new photos and the photo count on the devices was visible**.

Finally, it was evaluated if the photo count was an indication for the likeliness that users took advantage of such apps but **the data did not show any such relation exists**.

3.4 Summary

The average survey participant can be summarized as being 30 years old, owning an Android smartphone with 16 GB of storage, with 400 stored photos and ten stored videos. The user makes new photos regularly with the smartphone on a weekly basis and is not likely to backup or organize the media collection. Furthermore, if the user has a tablet it is very likely an iPad, which is rarely used for capturing but for viewing purposes. It has less media locally stored but uses cloud services extensively.

In comparison to the findings of [TGE11] in 2011, it was discovered that 70% (an increase of 25%) of the survey participants had more than 100 photos on their smartphones and 19% (an increase of 16%) had more than 1000 photos on their devices. Contrary to the belief that tablets deliver a better experience for viewing images and watching videos, participants in fact stored considerably less media on their tablets than in comparison to their smartphones. Furthermore, participants with iOS devices seem to have significantly more images stored on their devices than in comparison to participants with Android devices. It was also observed that in case of smartphones there was a significant difference between women and men. Female participants had significantly more images on their devices than men. Moreover, regardless in terms of the amount of images, users seem to rarely organize their photos in any way. This means there is potential for works that try to optimize the browsing process in such unmanaged collections.

As for user intentions and motives it was found that the most popular motives are non-surprisingly those for which the cameras have been optimized. Manufacturers of course know

since several years that people like to take photos of people and landscapes, at holidays and at events. However, the notion of document scanning and preserving notes and text with the camera seems to be a popular use case too, which has not yet been addressed at large. Focusing on the actual goal or intent of users it was discovered with the survey that not everything is meant for sharing. While lately most of the manufacturers and many of the researchers have focused on social sharing of images and automated image upload to the cloud, there is still the recognition of private and not-shared photos.

In accordance, the focus in this thesis will be to optimize the browsing experience for a collection of 100 to 400 images and around ten videos. Since ten videos is rather low it is reasonable to concentrate on browsing in a single video, at least for this work.

CHAPTER

4 Mobile Image Browsing on Tablets

Based on the insights of chapter 3, especially regarding the size of image collections, in this chapter, two image browsing interfaces for tablets are proposed and evaluated[1]. The interfaces not only explore a new kind of visualization but also utilize content analysis results. In the course of implementing and testing the interface ideas, it became clear that although smartphones and tablets have gained large performance gains over the last years, there are still limitations. Rodden et al. [RBSW01] showed that organizing images on similarity can have a positive effect on image browsing. As a result the following interface concepts all incorporate offline color sorting of images. Sorting images based on their dominant color was chosen because of the promising results shown by Schoeffmann and Ahlström [SA12b]. It was moreover already successfully applied to a mobile setting by Schoeffmann et al. [SAB11] and Ahlström et al. [AHSS12]. Furthermore, the described work in this chapter and chapter 5 raised the demand for a more thorough investigation of device performance in regards to content analysis. The results of this investigation can be found later in chapter 7.

All of the following interface concepts utilize the color sorting approach of Schoeffmann and Ahlström [SA12b]. Images are sorted based on their dominant hue level of the HSV color space. To do this, for each image a 24-bin HSV histogram is generated. Moreover, very bright and very dark images receive a special treatment. Since in such cases the dominant color becomes less important they are positioned differently in the sorted set of images. Very

[1]Please note that the contents of this chapter are adapted from [Hud13] and [HSA14]

bright images are placed at the beginning, whereas very dark images are placed at the end.

Based on earlier works of Schoeffmann et al. [SAB11] and Ahlström et al. [AHSS12] with following 3D interfaces it is investigated whether the shape of a 3D visualization actually makes a difference.

4.1 Color-Sorted 3D Image Browsing on Tablets

In the work by Ahlström et al. [AHSS12] two 3D image browsing interfaces for tablet devices were compared in a user study. Both used the shape of 3D globe. Moreover, they were contrasted with the performance of a color sorted grid image browser. The two globes (*HY-Globe* and *H-Globe*) differ in the way how the images are positioned on their surfaces. In the study the *H-Globe* was the most successful and could outperform its competitors.

In another work, Schoeffmann and Ahlström [SA12a] propose a color sorted 3D ring interface, which also performed significantly better when compared to a grid image browser. The interface was originally implemented and evaluated as a desktop application. Later, it was ported to a tablet device but not yet evaluated in that context.

The aim of the following work is now to find out whether there are differences in search performance between the Ring interface and the H-Globe. Both will use the same color sorting procedure as mentioned before. This precondition is necessary in order to be able to determine if the different shapes actually have an impact. Both interfaces will first be explained in greater detail. After that, setup and results of the user study are reported and discussed.

4.1.1 Color-sorted 3D Globe for Tablets

The work on the 3D Globe interface originally started with the intension to create a mobile, touchscreen-based version of the hierarchical 3D sphere interface of Schaefer [Sch10]. His interface places images on the surface of a rotatable 3D sphere based on their median H and S values of the HSV color space [SH98]. Furthermore, a grid-based clustering organizes the spheres' surface into image cells of equal size. Based on calculated surface coordinates (dependent on the median H and S values) images are assigned to the cells. To avoid holes

in this structure a distribution algorithm then shifts images to nearby empty image cells. This is done for a maximum of one quarter of a cells images. Image cells are visualized by one representative image. Other images of the cell can be explored by clicking on the cell, enabling a hierarchical tree-like browsing approach.

The touchscreen-based version that was implemented for an Apple iPad was published at the International Conference for Multimedia Retrieval 2012 [SA12a] and received good feedback. However, it became clear that the hierarchical approach can become quickly irritating for users, as they often had difficulties deducing to which image cell a desired image could have been assigned. As a result, alternative globe-based visualizations were investigated and contrasted with a simple 2D color sorted grid browser [Hud12, AHSS12]. For example, the *ZHV-Globe* that adjusts the amount of displayed image cells dependent on the current zoom level, the *HY-Globe* that displays image hierarchies via splitting of root image cells, removing the need to use representative images, or the *H-Globe* that uses a fixed image grid layout without image clustering or hierarchy generation. In the evaluation the *H-Globe* could outperform all other interfaces and is thus re-evaluated as *Globe* interface in this user study.

The Globe interface arranges images on a rotatable and zoomable 3D globe. The surface of the Globe is divided up into cells of same size, where each cell contains one image. The number of cells is calculated in advance to ensure that every image in the collection can be assigned. Moreover, the system tries to maintain an aspect ratio close to four-by-three to reduce visual distortion. Nevertheless, some distortion cannot be avoided, especially near the poles of the globe, given the nature of the surface.

In the configuration seen in Figure 4.1 the globe's cell lattice is made up of 12 cell rows and 30 cell columns, providing space for up to 360 images. The same configuration is used in the user study.

The color sorted images are assigned to the cells of the Globe in a one-by-one fashion. This is done by starting at a cell closest to the North Pole and then continuing south. To each cell an image is assigned on the way south until the cell closest to the South Pole is filled. When the whole cell column is complete the next column to the right is filled (column-major-order).

Figure 4.1: The Globe interface concept on an iPad with 350 images.

Users interact with the Globe by applying touch gestures like dragging, pinching or tapping. Drag gestures rotate and tilt the globe. In this way, the Globe can be rotated without limitations around its vertical axis. Tilting is limited to a maximum of 30 degrees in either direction. This restriction is required in order to avoid irritation of users when accidentally turning the Globe upside down. Zooming is realized with pinch gestures. Since the thumbnails can be too small to make out all the details of an image, this technique enables users to see less but larger and therefore more detailed thumbnails.

Finally, images can also be displayed in a *detail mode* by tapping on their thumbnail. The selected image is then displayed fullscreen. To return to the Globe a *back*-button can be used, which is displayed in the lower left corner of the screen.

4.1.2 Color-sorted 3D Ring for Tablets

This interface organizes the images in the shape of a 3D ring. Its organization is in fact very similar to the 3D globe: a grid of image cells that is wrapped in a hovering circle. Figure 4.2 shows the Ring providing five cell rows and 70 cell columns. This results in a capacity for up to 350 images and represents the configuration that was used for the evaluation. The color-sorted images are assigned by filling each column of image cells from top to bottom and then continuing with the next column to the right (counter-clockwise).

An apparent advantage of the ring-concept is that more thumbnails can be shown on the screen at once when it is slightly tilted to the front. Because of this, users are able to spot thumbnails that are currently at the back of the ring. As a result, more thumbnails are visible at once in comparison with the globe. The better overview is also helpful in case users want to scroll to a certain color. Since more of the actual color distribution is exposed it is easier to orientate and navigate. Moreover, thumbnails that are currently at the back of the Ring are automatically mirrored vertically. This prevents visual irritation of users, since otherwise the thumbnails would appear horizontally flipped. The mirroring process is hidden from users by performing it at the left and right sides of the ring, which are not visible. A slight black border around the thumbnails also visually separates them from each other. This should ease perception and avoid visual merging effects.

Users can interact with the Ring by applying dragging, tapping or pinching gestures. Rotating the Ring clockwise or counter-clockwise is possible by using horizontal drag gestures. Tapping once on one of the thumbnails changes the view to a *detail mode*, similar to what is used in the Globe interface.

The Ring interface also supports zooming the view by applying pinch gestures. The zooming process is more complex than with the globe. When zooming-in, the Ring reduces the tilting angle while at the same time increasing its size until the front part covers most of the available screen space. Additional zooming at that point results in transitioning the view to enable sight of the rings back area.

Figure 4.2: The Ring interface concept on an iPad at 350 images.

To directly zoom to the back area it is also possible to double-tap on the screen. Another double-tap returns the view to the default minimum zoom level. The idea behind this is that users can save time when they want a larger view of the thumbnails currently positioned in the back of the ring.

4.1.3 Evaluation

The aim of the evaluation is to find out whether one of the interfaces performs significantly faster in a KIS-like scenario (Known-Item Search). A user study was performed in which 16 participants took part (15 male and one female). All of the participants were members of our institute but none of them was involved in the project. All participants stated that they would work at least 40 hours professionally with a computer. Moreover, they were asked if they owned a smartphone or tablet to determine how much they were accustomed to touch

screen interactions. In total 15 participants were using a smartphone. Furthermore, four of the smartphone users told us that they would also use a tablet.

The user test was structured into two learning and two test phases. In the first learning phase participants had to read through an instructional text that informed them about what the study was about and how to use the first interface. When they were finished and had no further questions, they continued with the first test phase. In the following, they had to perform a number of trials with the first interface. After that they were asked to fill out a questionnaire and then continued with the learning and test phases of the next interface. In the end, after they completed the second questionnaire, participants had to fill out a final questionnaire that asked about their personal ranking of the interfaces, with the possibility to give further comments and feedback.

Since the same target images and the same image collection were used for both interfaces, the order in which the interfaces were tested was alternated between the participants. This was done to level out learning effects that might occur when participants memorized images from their first test phase.

In each of the two test phases the participants had to find a minimum of 40 *target images*, e.g., images that were shown to them and which they had to find with one of the interfaces. An image collection of 350 images was chosen based on the insights that were gained in chapter 3. The individual images were randomly drawn from the Wang image data set [WLW00], as well as a couple of key-frames from shots of the IACC.1 TRECVID 2010 repository [SOK06]. To further increase the diversity of the data set a number of randomly drawn images from Flickr were also included.

Each of the 40 trials started with the display of the target image in full resolution on the screen (see Figure 4.3 left). Text on the screen instructed the participants to memorize the image as good as possible. For this part they were allowed to take as much time as they wanted (there was no option to display the image again later). When they were sure they were ready, the participants could start the search phase with a tap on the *start*-button.

To complete the trial, participants had to tap on the appropriate thumbnail to switch the view into *detail mode* and tap on the *accept*-button (see Figure 4.3 right). Furthermore, participants had the possibility to return to the browsing interface without submitting the image via a *back*-button.

After submitting an image, participants got immediate feedback whether their choice was correct or not. This was realized by turning the whole screen green (for correct) or red (for incorrect) for a few seconds. After that, they were transferred to the next trial.

Participants also had the possibility to abort a trial in case they forgot how the desired target image looked like, or if they wanted to give up on searching. For this reason, a *skip*-button was always visible during the search process. Tapping this button would result in canceling the current trial. Participants would then be immediately transferred to the next trial.

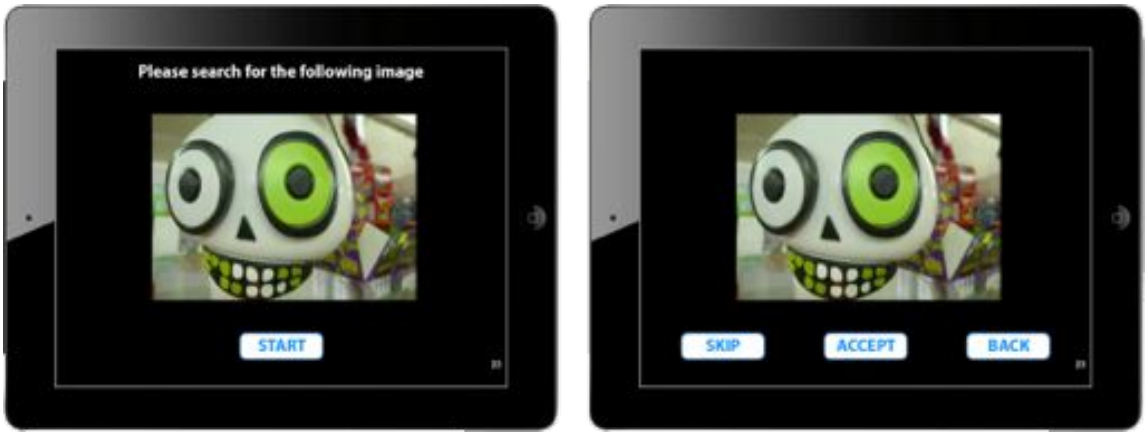

Figure 4.3: Trial start (left) and detail mode (right) for both interfaces.

4.1.4 Target Groups

In this study so called *target groups* are used. They are intended to even out different search difficulty levels between images. This concerns the problem that, due to their contents, some images might be easier to spot than others. The subjective performance of each individual participant at a specific day and time also plays a part in this matter. Therefore, images in a certain area are logically grouped together to form a target group. Any image in such a

group should (in theory) require a similar amount of search effort and could be exchanged with any other image of the same group. This makes it possible to provide alternatives to skipped images that require a similar amount of search effort.

All images of the data set were grouped into ten target groups, such that each group included 35 images. The groups were formed by dividing the color-sorted list of images into ten groups of equal length. Since the images are applied in both interfaces in a one-by-one manner to the cells of the 3D surfaces, images that are similar in color are also spatially close to each other in the interfaces. See figure 4.4 for an example.

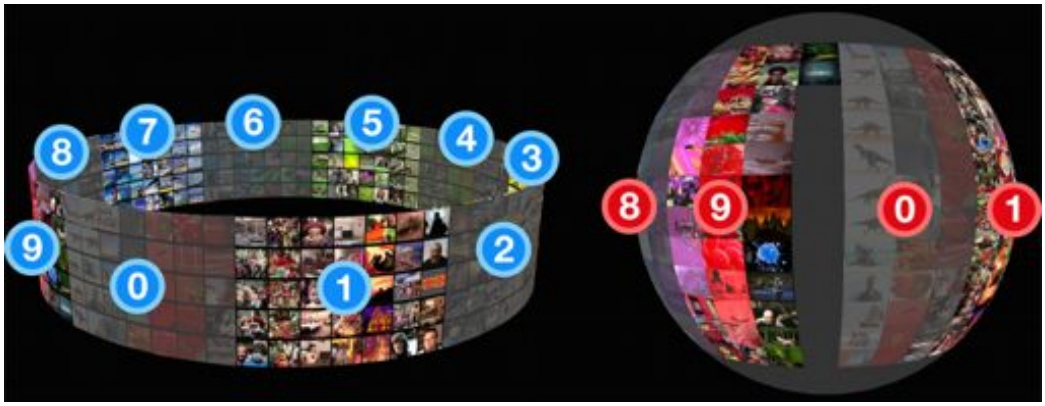

Figure 4.4: Target groups marked in Ring interface (left) and Globe interface (right).

As already mentioned, target groups are used to appropriately react to skipped images. To do this two lists are created: list A and list B. List A defines in which order new images are drawn from ten image pools. The images contained in each pool are described in list B. For each image pool seven images are randomly drawn from one of the ten target groups (each containing 35 images). The first four images of each pool are considered as default target images. This will result, in case no images are skipped, in a total of 40 target images being displayed. The additional three images serve as backup in case participants choose to skip one or more images of the pool. In such a case users continue with the next image out of the same pool. If users keep skipping images, this process is continued until all seven

images have been used. In that case the participants continue with images of the next pool defined in list A until all image pools are depleted.

The lists were randomly created in advance. All participants used the same lists to maximize comparability. Due to the explained procedure of trial management it was expected that not all participants performed the same amount of trials.

4.1.5 Questionnaires

In the questionnaires participants had to rate the interfaces and give comments about their personal subjective experience. They could also suggest improvements if they thought of any. The questionnaires were designed in the style of the NASA TLX workload index [HS88] and remained the same for both interfaces. Nine questions had to be answered on a Likert scale rating, ranging from 1 to 10. The questions were about *Mental demand*, *Physical demand*, *Experienced temporal pressure*, *Effort*, *Frustration*, *Fun*, *Experienced support of interface visualization technique*, *Learning effect* and *Experienced support of color sorting*. A value of one was the best achievable value for all questions except for *Fun*, *Experienced support of interface* and *Experienced support of color sorting*.

In the final questionnaire at the end of the user test, participants additionally had to report which of the interfaces they would personally prefer (by assigning a first and a second rank). Furthermore, they were able to give final remarks and comments about the whole user test experience.

4.1.6 Results - Trial Distribution

In total 691 trials were performed with the Globe interface. 618 trials were correct (i.e., participants submitted the right image), 54 were skipped and 19 were wrong. With the Ring interface 678 trials in total were performed. 604 were correct, 39 were skipped and 35 were wrong submissions. As mentioned before, the trial counts are different between the interfaces because participants could freely choose to skip target images.

4.1.7 Results - Trial Times

In this analysis it is evaluated whether there are statistical significant differences for the following factors: (1) *interface* (i.e., differences between interfaces), (2) *target group* (i.e., differences between target groups across all interfaces), and (3) *interface* $\times$ *target group interaction* (i.e., differences between combinations of the two factors). Additionally, partial eta-squared (η^2) is reported, which is a measure of effect size. It can be interpreted as follows: 0.01 signifies a small effect size, 0.06 signifies a medium effect size and 0.14 signifies a large effect size, as described by Cohan [Coh73].

The statistical analysis of trial times is based only on correct trials (participants found the correct image). As search times were positively skewed a logarithmic transformation was applied to achieve a near normal distribution, as it is a requirement to perform a statistical ANOVA test. For that reason, geometric mean search times are reported, which are the result of an anti-logarithmic transformation of calculated mean values in the log scale. For the Globe interface the geometric mean search time was **9.42** seconds, for the Ring interface it was **10.61** seconds. To determine statistical significance of the difference a repeated measures ANOVA was applied with the independent factors *interface* and *target group*. The test did not show any statistical significant main effect for *interface* ($F_{1,15} = 3.07, p = 0.1, \eta^2 = 0.17$), but a statistical significant main effect for *target group* ($F_{5,74.2} = 40.10, p < 0.0001, \eta^2 = 0.728$; Greenhouse-Geisser corrected) and a statistical significant *interface* $\times$ *target group* interaction ($F_{4.1,61.2} = 6.22, p < 0.0001, \eta^2 = 0.293$; Greenhouse-Geisser corrected).

As can be seen in Figure 4.5, there are major deviations between the target groups *one* and *nine* of the two interfaces. After analyzing the configurations of both interfaces thoroughly it can be concluded that the difference primarily was caused by a slight different initial rotation of the interfaces.

In Figure 4.4 it is visible that target group *one* was slightly better exposed at the trial start on the Ring interface than what was the case for the Globe interface. Target group *nine* on the other hand was under-exposed on the Ring interface accordingly. It can be presumed that participants therefore were able to find images placed in target group *one* much faster with the Ring interface. Moreover, images of target group *nine* should have

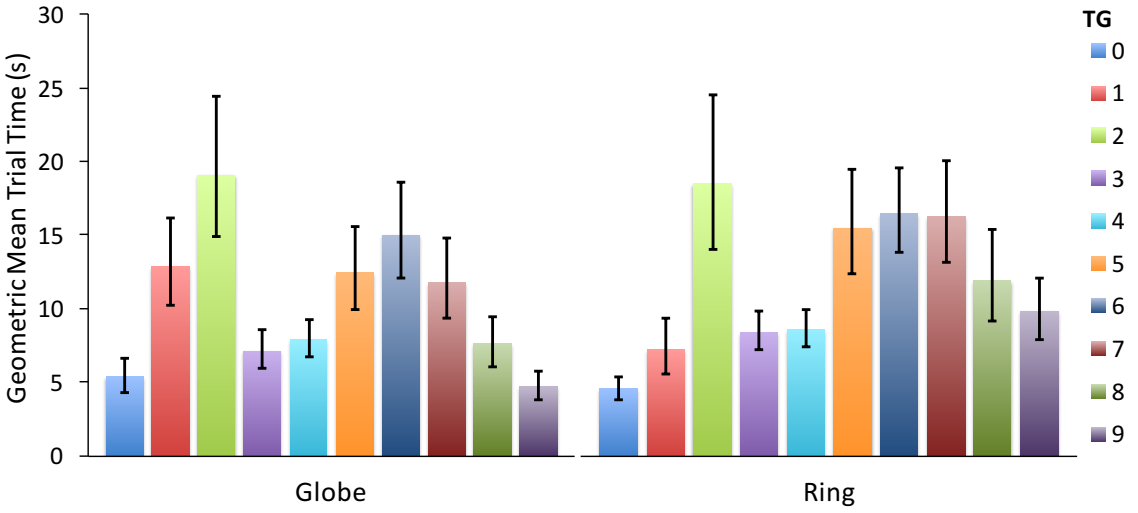

Figure 4.5: Trial times between target groups of Globe and Ring (Error bars: 95% CI).

be been much easier to find with the Globe interface. All other differences that could be measured between the interfaces were statistically not significant.

4.1.8 Results - Questionnaires

Ratings for both interfaces were very similar as can be seen in Figure 4.6 (lower is better except for *Fun*, *Support of Interface* and *Support of Color Sort*). For each of the questions a t-Test was performed. No significant difference between the interfaces could be measured. This also was reflected in the subjective interface rating of the final questionnaire. The Globe and Ring interface scored equally well with both interfaces getting ranked at first place by eight participants.

4.1.9 Results - Summary

As reported in the analysis, no significant differences could be found except for the target groups *one* and *nine*, which is negligible as explained earlier. The mentioned anomalies most likely will not have a big impact on search performance in real life. When considering the results of Ahlström et al. [AHSS12] and Schoeffmann and Ahlström [SA12a], it seems that

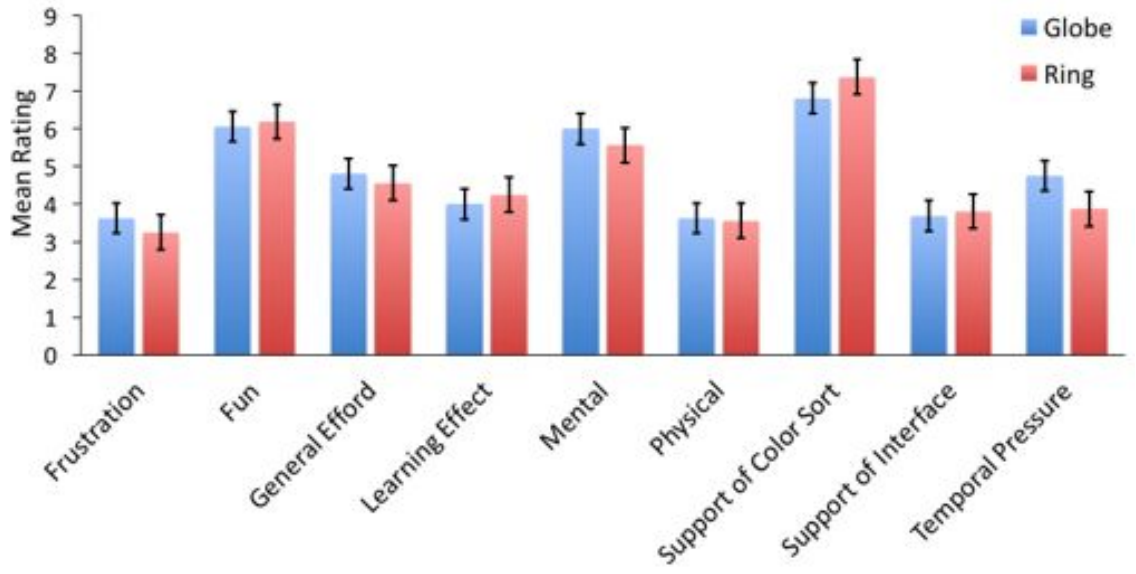

Figure 4.6: Questionnaire ratings between Globe and Ring (Error bars: +/- SE).

color-sorted 3D interfaces can show an improvement in comparison to typical (color-sorted) 2D grid layouts but it does not matter if a ring-like or a globe-like visualization is used (at least when using a tablet).

These findings are further supported by how the participants ranked the interfaces in the questionnaires. No significant differences for the answers of the regular questions and an exact tie in terms of interface ranking shows that also in subjective terms the interfaces were similarly easy to use.

This result is interesting, as it was actually expected that the Ring would outperform the globe. As mentioned before, the Ring actually exposes more images on the screen to users since also a large part of images currently at the back of it are visible. Participants seem not to have taken advantage of this feature at all. A possible reason for this could be that users need more time to get used to the interface in order to be able to utilize the rings' functionality to its fullest. Another reason could be that the thumbnails at the back are already quite small for viewing and interaction. Therefore, users may concentrated primarily on the front part of the ring, which would result in a similar scrolling effort to what is necessary with the globe.

CHAPTER

5 Mobile Image Browsing on Smartphones

The earlier reported study concentrated on *tablet devices* for image browsing. The next step is a transition to smaller devices like smartphones, and thus smaller screens. The greatly decreased screen size could potentially influence the performance of 3D interfaces, since they cannot utilize available screen space as efficient as 2D interfaces. The 3D Ring and 3D Globe interfaces are adapted to these smaller screen sizes (regarding column and row setup) and re-evaluated. Furthermore, the Ring is extended with small interaction improvements.

Since both interfaces were never compared in earlier studies to a color-sorted standard 2D grid at this screen size before, such an interface type is also added to the mix. Moreover, another color-sorted 2D interface idea is proposed - the *ImagePane*.

All four interfaces again use the same color sorting algorithm that is applied by the tablet-centric prototypes [SA12b]. This time, the interface concepts were implemented on a fourth generation Apple iPod Touch with Objective-C and OpenGL ES. The iPod Touch was chosen because it has the same screen-size (3.5 inches) and similar performance to the (at that time) very popular iPhone 4/4s while keeping cost low.

Furthermore, the interfaces are tested with four different image data sets, ranging from 100 up to 400 distinct images, as can be seen in Figure 5.1.

In the following, the adjustments that were necessary in terms of Ring and Globe are described, followed by an introduction to the Grid and the ImagePane interface.

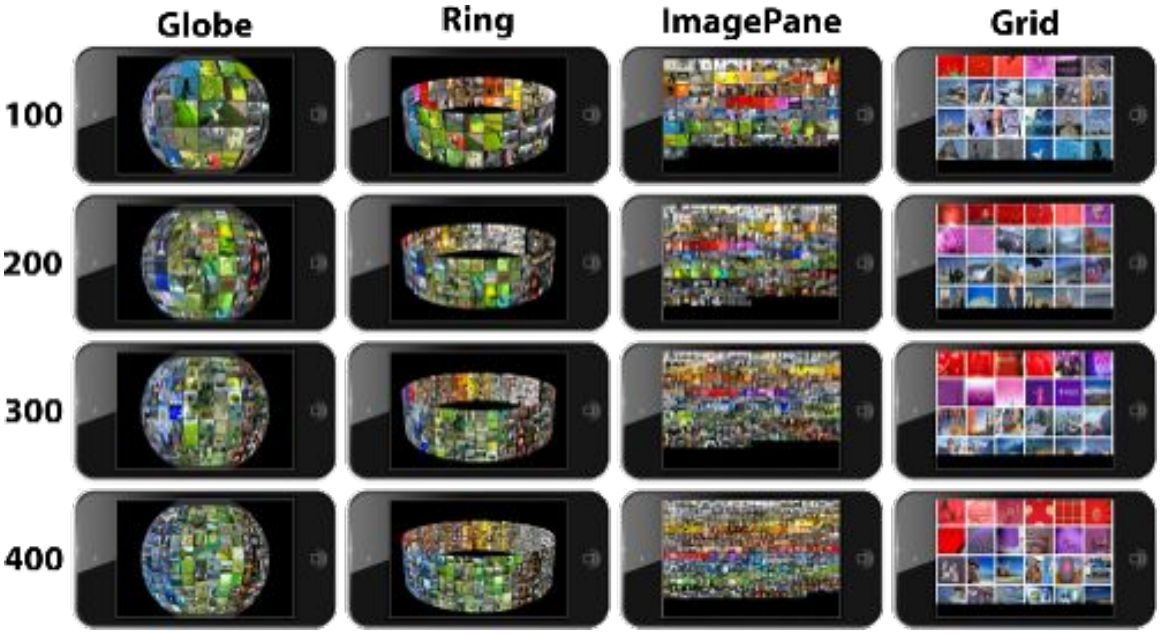

Figure 5.1: Smartphone interfaces in configurations with 100, 200, 300 and 400 images.

5.1 Globe

The Globe interface is adjusted in terms of column/row counts, rotation speed and maximum zoom-level. Although columns and rows adjust automatically to minimize image distortion, some slight changes in respect to the now largely reduced screen space had to be implemented. Particularly, it had to be ensured that the hit targets of each thumbnail image were large enough for individually tapping them. Moreover, it is necessary to make some changes to the code that managed the rotation of the globe. Since the finger movement is much shorter when swiping from left to right on smartphone screens, the globes rotation speed is increased in order to make the interaction feel natural again. The same also applies to the tilting mechanism.

Last, the maximum zoom level is adjusted as due to the new environment the current configuration would have caused users to be able to actually scroll through the globe. Everything else remains the same to the earlier tablet-optimized version.

5.2 Ring

The smaller version of the Ring is altered in a similar way as the Globe interface (rotation speed, column/row adjustments, etc.). Additionally, a number of improvements that were often mentioned by previous test participants are introduced. First, the height and the position of Ring are changed slightly to better use the available screen space. The original version is not completely centered vertically, which wastes screen real estate. Second, the gaps between the images are changed. In the earlier version users are able to see through this gaps. This was reported to be very irritating by many participants. As a reaction, in this version the gaps are realized with an opaque black border.

Another frequent suggestion by participants concerns the sensitivity in which the Ring registers selective single-taps in contrast to drag gestures for rotation. This often resulted in undesired image selection. To solve this problem the thresholds in the gesture recognition algorithms are changed.

Furthermore, a new gesture is implemented. When users perform a swipe-up on the screen the interface directly zooms to the front of the ring, filling most of the screen space, as can be seen in Figure 5.2 on the left side. A subsequent swipe-down resets the interface again.

Additionally, double-tapping is improved by taking the tapping location into account. The Ring is first rotated so that the touched area is centered at the back. After that, the actual zoom operation to the back is performed. Both steps happen so fast that users are not able to notice it. This new behavior is meant to save users time. Originally they first had to zoom and then manually adjust the rotation of the Ring appropriatly. Please see Figure 5.2 (right) for reference.

5.3 ImagePane

The idea behind this interface is quite simple: give users all the information at once and let them choose which area to explore in more detail. It was born out of the need to maximize the usefulness of the color sorting. To do that, users should be able to instantly access

Figure 5.2: Zoomed view to the front (left) and back (right) of the Ring interface (100 images).

images of a given dominant color. Another objective was, to re-evaluate the impact of the 3D interfaces by contrasting them with an improved 2D approach that utilizes the same color sorting method.

The ImagePane displays all images of the collection at once. Small thumbnails represent each image. As a result, all color regions are immediately visible and accessible. This visualization of course has its limits. It cannot be used to inspect images in detail, as the thumbnails are too small. Users are also not expected to select individual images in this view, as the hit targets are very small too (although this functionality is available). It is meant more to give a first lead as to where the wanted image might be located in the collection.

The image placement is implemented in a very simple way. The color sorted images are applied left to right, row after row. Even with this simplistic approach the color sorting effect is visible, as can be seen in Figure 5.1.

Users can narrow down their search by zooming the view to a specific region. This can be done by performing a double-tap on the wanted area. In that zoomed mode the quadratic thumbnails are big enough for more thorough inspection. Furthermore, users are not locked to that zoomed area. It is possible to scroll through the whole pane vertically as well as horizontally. Finally, to display images in their full size users have to single-tap on the appropriate thumbnail.

The ImagePane adapts automatically to the amount of images in the collection. This is realized by a simple measure: the more images there are, the smaller the quadratic thumbnails become. That means that with increasing collection sizes the importance of the color sorting raises, as it becomes very hard to recognize details or select individual thumbnails.

5.4 Grid

The standard Grid interface is designed to look similar to the pre-installed photo browser on iOS 7 devices. The only enhancement is that it uses the same color sorting as the other interfaces.

The thumbnails do not have the same aspect ratio as the original image. For space reasons all thumbnails are quadratic in size. The contents of the original images are not distorted but are zoomed and cropped while preserving their aspect ratio.

The interface displays six thumbnails in each row in landscape orientation. Moreover, three and a half rows of thumbnails are visible at any time on the screen. Please see Figure 5.1 for reference. The user can scroll up or down by using swipe gestures. Images for display in their full size are selected by tapping on the appropriate thumbnail.

5.5 Evaluation Setup

In contrast to the tablet-centric user study, the interfaces this time were evaluated with four different image collections that differed in terms of images and in terms collection size. The four sets were configured with 100, 200, 300 and 400 images in order to see how the interfaces would perform when they had to visualize more and more images. The sizes also go along with what we discovered in chapter 3. The Wang [WLW00] and TRECVID [SOK06] datasets as well as Flickr served as basis from which images were randomly drawn.

Moreover, the evaluation consisted of two parts: an analysis of navigation support by performing a user interaction simulation and a between-subject user study.

5.6 Evaluation by User Simulation

In addition to a user study, a user simulation similar to works of other authors in the field (e.g., [Sch10], [STF+12] or [NW08]) was performed. Such simulation methods can reflect real user behavior only up to a certain degree and should not be used as sole evaluation methods. Nevertheless, they can be seen as complementary assessment and are an easy and fast way to get a rough estimate of how well an interface might perform compared to other interfaces.

It has to be noted that the following interaction simulation is assumed to operate under optimal conditions. Simulated users have perfect vision and understanding of the interfaces, e.g., are able to correctly identify the dominant color of an image and know how to use the color sorting for their advantage. It is also assumed that users recognize the thumbnail of the wanted image as soon as it is visible on the screen in a size of four by four millimeters. This is based on findings in works by Torralba [TFF08] and Hürst [HSST10]. Furthermore, after the recognition they immediately select the right image for display in detail mode and accept it by pressing the *Accept*-button.

Five different interaction categories are defined and the following amount of *interaction points*, which represent the required effort, are assigned to each one:

Gesture	Interaction Points
Single-Tap (st)	1
Double-Tap (dt)	1.5
Swipe (sw)	2
Drag (dg)	2.5
Pinch (pn)	3

Table 5.1: Definition of gestures and their interaction points.

For this simulation the gestures have to be defined thoroughly. A single-tap gesture (st) is a single quick tap on the screen with one finger. A double-tap gesture (dt) is done with two single-taps quickly performed after each other at approximately the same area on the screen. When performing a swipe gesture (sw) users place a single finger on the screen and

quickly move it in one direction without losing the contact to the screen. For this gesture users do not decelerate their finger before lifting it off the screen. A drag gesture (dg) is the same as a swipe gesture but it is usually performed slower and users decelerate their finger until it stops before they lift it off the screen. Finally, a pinch gesture (pn) is defined as placing two fingers on the screen (e.g., index finger and thumb) and moving them closer to each other or apart, without losing the contact to the screen.

The weighting seen in Table 5.1 is used to reflect how much physical interaction is necessary to perform one interaction step of a category. For a given goal (e.g., finding an image) the needed interaction steps are counted, weighted and summed to get an overall interaction score for an interface.

Figure 5.3: Target groups of the small versions of the Ring (left) and Globe (right) with 300 images.

To see how the interaction scores change between different target positions, four interaction groups are defined that span over a set of spatially close target groups. Since the images in the interaction groups are in approximately the same area, they should require the same amount of interaction. Furthermore, it is simulated that users search for a target image that is placed in the center of each interaction group.

The following groups are defined:

- **Interaction group one**: target groups 0 - 2
- **Interaction group two**: target groups 3 - 4
- **Interaction group three**: target groups 5 - 7
- **Interaction group four**: target groups 8 - 9

For a visualization of the target groups see Figure 5.3. In the following the interaction demands for the four interaction groups in the case of every interface is reported.

5.6.1 Interaction Group One

For the sets of 100 and 200 images, the image pane, Globe and Ring are very similar in their interaction requirements. The desired thumbnail is already displayed on the screen (see Figure 5.1). Therefore, a single-tap is already sufficient to show the target image in detail mode. Also for set sizes 300 and 400 the interaction needs are identical. All interfaces require that users perform a zoom operation because the thumbnails are too small for immediate identification of the right one. After zooming, they are able to select the right thumbnail with a single tap on the screen. The zooming operations of the interfaces require different kinds of interactions. In case of the ImagePane a double-tap is required. The Ring interface requires users to perform a swipe-up gesture to zoom to the front part of the ring. Finally, pinching is required with the Globe interface.

When comparing the interaction needs of the interfaces it can be seen that the Grid interface shows a positive correlation with the set sizes. The more images the more scrolling is needed. With 100 images no further interaction is necessary other than a tap on the already displayed thumbnail. At 200 images users first have to apply a drag gesture in order to scroll to the thumbnails location. A collection of 300 images requires two additional drag gestures and with 400 images three additional drag gestures are needed. A summery of all interaction scores for this group can be seen Table 5.2.

Interface	100	200	300	400	Avg.
ImagePane	1(st)	1(st)	1.5(dt) + 1(st)	1.5(dt) + 1(st)	1.75
Globe	1(st)	1(st)	3(pn) + 1(st)	3(pn) + 1(st)	2.5
Grid	1(st)	2.5(dg) + 1(st)	2 * 2.5(dg) + 1(st)	3 * 2.5(dg) + 1(st)	4.75
Ring	1(st)	1(st)	2(sw) + 1(st)	2(sw) + 1(st)	2

Table 5.2: Required interactions for interaction group one.

5.6.2 Interaction Group Two

At interaction group two the ImagePane requires the same amount of interaction with 100 and 200 images as what is the case at interaction group one. The Globe and Ring interfaces require one additional drag gesture to perform rotations to make the wanted thumbnail visible. When the collections of 300 and 400 images are used, users additionally have to zoom the view to make the thumbnail large enough. The Grid interface requires users to apply two drag gestures (100 images), three drag gestures (200 images), four and a half drag gestures (300 images) and seven drag gestures (400 images) in addition to a single tap to select the wanted image. All interaction scores are summarized in Tables 5.3 and 5.4.

Interface	100	200
ImagePane	1(st)	1(st)
Globe	2.5(dg) + 1(st)	2.5(dg) + 1(st)
Grid	2 * 2.5(dg) + 1(st)	3 * 2.5(dg) + 1(st)
Ring	2.5(dg) + 1(st)	2.5(dg) + 1(st)

Table 5.3: Required interactions for interaction group two (part one).

Interface	300	400	Avg.
ImagePane	1.5(dt) + 1(st)	1.5(dt) + 1(st)	1.75
Globe	2.5(dg) + 3(pn) + 1(st)	2.5(dg) + 3(pn) + 1(st)	5
Grid	4.5 * 2.5(dg) + 1(st)	7 * 2.5(dg) + 1(st)	11.3
Ring	2.5(dg) + 2(sw) + 1(st)	2.5(dg) + 2(sw) + 1(st)	4.5

Table 5.4: Required interactions for interaction group two (part two).

5.6.3 Interaction Group Three

The interaction effort for the ImagePane remain the same as in the earlier described interaction groups.

In case of the Ring interface one tap is sufficient with sets of 100 and 200 images. Interaction group three is already exposed in a sufficient size on the screen, therefore no additional scrolling or zooming is required. When used with sets of 300 and 400 images, users first need to use a double-tap gesture to zoom to the back part of the ring. After that they can select the target image with a single tap.

The Globe interface requires multiple drag gestures as interaction group three is placed on the back of it. In detail, it requires for the sets of 100 and 200 images two drag-gestures, either to the left or the right, plus a tap on the screen to select the appropriate image. Furthermore, in case of the sets with 300 and 400 images an additional pinch is needed to scale the thumbnails to a convenient size.

The Grid interface follows the trend seen in interaction groups one and two. The interaction effort increases in relation with the number of images in the set. At 100 images users have to apply three drag gestures to scroll the Grid to the appropriate position. A collection of 200 images already demands six drag gestures. In case of the set with 300 images, ten drag gestures are required and with the largest set of 400 images 14 drag gestures have to be used. Moreover, an additional tap is needed to select the desired image. Tables 5.5 and 5.6 summarize all interaction steps for all interfaces/sets.

Interface	100	200
ImagePane	1(st)	1(st)
Globe	2 * 2.5(dg) + 1(st)	2 * 2.5(dg) + 1(st)
Grid	3 * 2.5(dg) + 1(st)	6 * 2.5(dg) + 1(st)
Ring	1(st)	1(st)

Table 5.5: Required interactions for interaction group three (part one).

Interface	300	400	Avg.
ImagePane	1.5(dt) + 1(st)	1.5(dt) + 1(st)	1.75
Globe	2 * 2.5(dg) + 3(pn) + 1(st)	2 * 2.5(dg) + 3(pn) + 1(st)	7.5
Grid	10 * 2.5(dg) + 1(st)	14 * 2.5(dg) + 1(st)	21.6
Ring	1.5(dt) + 1(st)	1.5(dt) + 1(st)	1.75

Table 5.6: Required interactions for interaction group three (part two).

5.6.4 Interaction Group Four

The ImagePane implies the same amount of interaction as with the earlier groups.

The Ring and the Globe interface require one drag gesture for right rotation as well as a tap for sets with 100 and 200 images. The larger sets of 300 and 400 images require after a drag gesture an additional swipe in case of the Ring and a pinch in case of the Globe interface to zoom the view appropriately.

The analysis reveals that interaction group four is the most problematic group for the Grid interface. Paired with the set of 100 images it requires four drag gestures to scroll to the needed area. At 200 images it already requires nine drag gestures and 300 images demand the Grid to be scrolled by using 14 drag gestures. Finally, the largest set of 400 images requires users to apply 19 drag gestures.

All interaction steps are summarized in the Tables 5.7 and 5.8.

Interface	100	200
ImagePane	1(st)	1(st)
Globe	2.5(dg) + 1(st)	2.5(dg) + 1(st)
Grid	4 * 2.5(dg) + 1(st)	9 * 2.5(dg) + 1(st)
Ring	2.5(dg) + 1(st)	2.5(dg) + 1(st)

Table 5.7: Required interactions for interaction group four (part one).

Interface	300	400	Avg.
ImagePane	1.5(dt) + 1(st)	1.5(dt) + 1(st)	1.75
Globe	2.5(dg) + 3(pn) + 1(st)	2.5(dg) + 3(pn) + 1(st)	5
Grid	14 * 2.5(dg) + 1(st)	19 * 2.5(dg) + 1(st)	29.8
Ring	2.5(dg) + 2(sw) + 1(st)	2.5(dg) + 2(sw) + 1(st)	4.5

Table 5.8: Required interactions for interaction group four (part two).

5.6.5 Summary

Table 5.9 shows the average interaction scores for each interface over all image sets and interaction groups.

Interface	Avg. Interaction Points
ImagePane	1.75
Globe	5
Grid	16.9
Ring	3.2

Table 5.9: Average interactions needed with each interface (lower is better).

Under the earlier described simulation conditions, the ImagePane performs best of all interfaces. It provides direct access to all images in the collection from the start and examining a certain color region in greater detail is achieved by a simple double-tap. Furthermore, it turns out that the second best interface is the 3D ring. It also exposes almost the whole image collection with only little additional interaction effort to reveal the rest. Third rank can be assigned to the 3D Globe interface. The Globe primarily lost to the Ring because of the extra interaction need to reach images that are placed at the back.

The interface that requires most interaction is the traditional color sorted Grid interface. Since it cannot display as many images as the other interfaces, it has a major disadvantage. The situation becomes even worse the more images are in a collection. The result of interaction groups three and four are a good example of this problem. They require users to apply over ten consecutive drag gestures on the screen to get to the appropriate area.

Although the results of this simulation have to be taken cautiously they already show one of the major problems of traditional grid interfaces. The more images there are in the collection and the lower the searched item is located in the grid, the more interaction is needed.

The exceptional good performance of the ImagePane on the other hand is rather theoretical, as most likely actual users' vision and perception will not be that perfect, so that they can reproduce the results in all cases.

5.7 Evaluation - User Study

In total 48 participants took part in the study, of which 23 were male and 25 female. Average age was around 24 years and the self-reported weekly computer usage was around 40 hours a week. All but four participants used a smartphone with a touchscreen and 16 had used or owned a tablet. Three quarters of the participants were students of social or psychology sciences. One fourth was recruited from members of our institute that were not involved in the project. The participants were grouped in four groups of twelve, assuring that three in every group were institute members. This setup should balance out novice and more professional users. Each of the groups tested only one interface, but with four different image sets (between subject study design).

Testing an interface with an image set consisted out of finding 60 target images. For each of the image sets a list of 60 randomly drawn target images was created. The lists remained the same for all participants and for all interfaces. In contrast to the tablet user study, participants were not allowed to skip target images. Instead they had to find the target image within one minute. After that, the target image was automatically skipped and the participants were transferred to the next target image. This was done because it was discovered in the tablet user study that participants would skip the image after one minute anyway. On the other hand, in this way it could be avoided that participants would invest too much energy in a single image.

Searching all 240 target images in one single session would be too much of a strain. Therefore, the study was designed as a two-day assignment. Participants could freely choose two days within a week. On the first day they would receive the initial introduction and would perform the test with the first two image sets, finishing the first day with filling out a questionnaire. On the second day they would continue with the remaining two image sets, fill out a second questionnaire with the same questions and give their final remarks. The order in which the sets were tested was alternated between the participants.

Before starting with the first session participants had to read an explanatory text about the goals of the user study and how their assigned interface is used. This was followed by a training session. When participants said they were comfortable with using the interface the test session was started.

A single trial was structured as follows: The target image was displayed on the screen and participants could take as much time as they wanted to memorize it. When they felt ready they started the trial by a single tap on the *Start*-button. They were then transferred to the browsing interface and continued with searching for the image. When displaying an image in *detail mode*, participants were able to submit the image by tapping the *Accept*-button or return to the browsing interface by using the *Back*-button. Submitting an image ended the trial by turning the whole screen green (correct image) or red (wrong image). Moreover, if participants took longer than one minute the trial was also ended by turning the screen red. After that they were transferred to the target image of the next trial until all trials had been completed.

5.8 Results - Trial Distribution

Over the span of the whole study the participants performed 11520 trials. Each of them performed 240 trials with their assigned interface. This means that each interface was tested by 2880 trials in total. Of those, the Globe interface showed 2505 correct trials (87%), 295 skipped (10%) and 80 wrong (3%) trials (users submitted the wrong image). The Ring interface showed 2452 correct trials (85%), 287 skipped trials (10%) and 152 wrong trials

(5%). The ImagePane showed 2449 correct trials (85%), 279 skipped trials (10%) and 152 wrong trials (5%). Eventually, the Grid interface had 2696 correct trials (94%), 142 skipped (5%) and 42 wrong (1%) trials.

5.9 Results - Trial Times

In the statistical analysis only correct trials are considered (trials completed with the correct image). The search times were positively skewed, which is why a log-transformation was applied. After the transformation the data showed a close to normal distribution. In the statistical analysis the geometric mean search time is reported. The Grid interface showed a geometric mean search time of **8.04** seconds, Globe as well as ImagePane of **8.12** seconds and the Ring showed a geometric mean search time of **9.25** seconds. See Figure 5.9 for a visualization of the data.

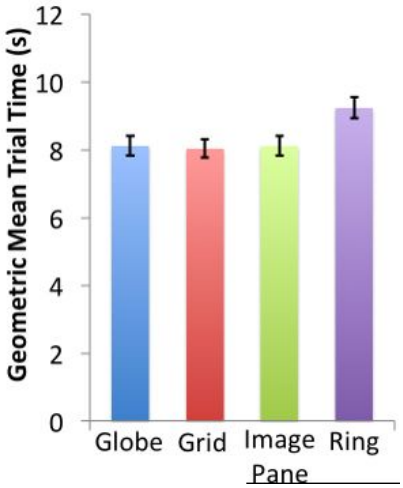

Figure 5.4: Geometric mean trial times of all smartphone interfaces.

A mixed ANOVA with repeated measures was utilized to detect statistical significant differences. Independent factors *set size* and *target group* (inside subjects) and *interface* (between subjects) showed no significant main effect for *interface* ($F_{3,44} = 2.71, p = 0.056, \eta^2 = 0.156$) but a significant main effect for *set size* ($F_{3,132} = 398.94, p < 0.0001, \eta^2 = 0.901$), *target group* ($F_{5.7,252.6} = 179.81, p < 0.0001, \eta^2 = 0.803$; Greenhouse-Geisser corrected) as well as *set size* × *interface* interaction ($F_{9,132} = 4.51, p < 0.001, \eta^2 = 0.235$), *target group* × *interface* interaction ($F_{27,396} = 23.54, p < 0.0001, \eta^2 = 0.616$), *set size* × *target group* interaction ($F_{16.2,710.9} = 22.66, p < 0.0001, \eta^2 = 0.34$; Greenhouse-Geisser corrected) and *set size* × *target group* × *interface* interaction ($F_{81,1188} = 1.61, p = 0.001, \eta^2 = 0.099$).

In Figure 5.5 it can be seen that the Grid interface shows a steady increase of search time in relation to the set size, which is not the case for the other interfaces. In fact, the globe, Ring and ImagePane show more of a leveling down at the larger set sizes. It would be interesting to do further research if this trend continues with set sizes of 500 and more images.

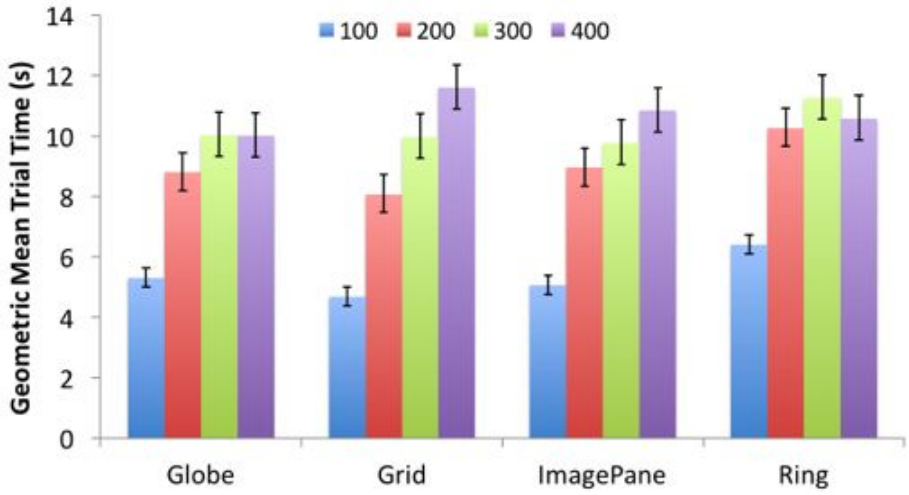

Figure 5.5: Geometric mean trial time per set per interface for smartphones (Error bars: 95% CI).

The geometric mean search times are shown in contrast to the target groups in Figure 5.6. As can be expected for the Grid interface, the lower the target group is placed in the Grid the higher the mean search time (with target group one at the top and target group ten at the bottom of the grid). Also, color sorting of the images did not help in this case. In contrast, the other interfaces do not show such a linear increase. The geometric mean search times are much more evenly distributed across all target groups. Especially the ImagePane seems to provide close to equal access time to all ten target groups.

In Figure 5.6 it is also visible that the Ring and the Globe interface have a common weak point at the target groups *three* and *nine*. At trial start, these two target groups are located in both interfaces at the far left and the right sides and are therefore barley visible. This can easily be seen when looking at Figure 5.3. What is interesting is that although the images at the back of the Globe are not visible, participants needed much less time to find

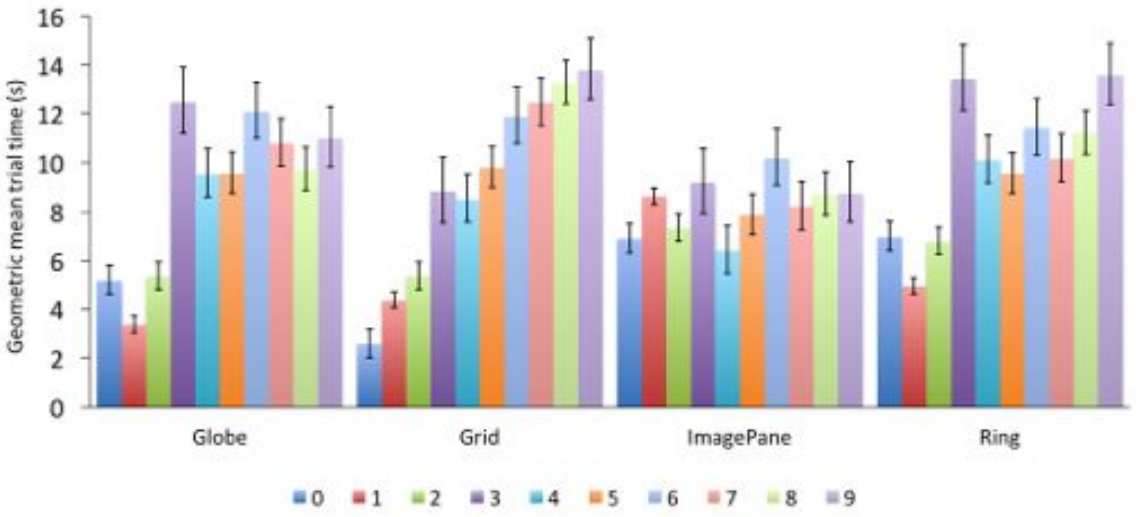

Figure 5.6: Geometric mean trial time per target group and interface (Error bars: 95% CI).

those than the ones that resided at the edges. Most probably, the reason for this is that a little more than a single drag gesture across the whole screen already turned the Globe by a full 180 degrees. The participants would therefore first inspect the front, then the back and then continued with the edges.

To avoid this problem it might be best to prevent placing images at those edges altogether. The two interfaces could be redesigned as follows:

The Ring interface could be split into two slightly less curved half-circles or bands. During scrolling images would transition from one band to the other seamlessly. Another, more simplistic idea would still show the illusion of a full ring, but images would avoid the edges when scrolling and directly transition to the back or the front of the ring. With this redesign all images would be visible all the time.

A redesign of the Globe is more difficult. The same approach would not work, as images that disappear on the right side would immediately show up on the left side again. This could cause very irritating visual effects for users. As an alternative, two views of the same

Globe could be visualized. One would show the front, the other would show the back of it. When users scroll one of the views the other one would be scrolled accordingly. A downside of this visualization could be that users perception could be overwhelmed and therefore only look at one of the globes. This would render the other one useless.

5.10 Results - Questionnaires

The questionnaires followed the same style as used in the tablet user study. This is also true for the asked questions. A mean is calculated for the questions of both user study days, to level out day-dependent preferences. The mean scores for each question and interface can be seen in Figure 5.7. For each question a Kruskal-Wallis test was performed, which showed no statistical significant difference except for *Effort* ($\chi^2(3) = 9.3, p = 0.026$) and *Frustration* ($\chi^2(3) = 8.4, p = 0.039$). Pairwise comparisons using Dunn's procedure [Dun64] with a Bonferroni correction for multiple comparisons showed significant differences for *Effort* between Grid (Mdn = 5.0) and ImagePane (Mdn = 7.0)($p = 0.029$) and for *Frustration* between Grid (Mdn = 5.0) and Globe (Mdn = 7.0)($p = 0.036$).

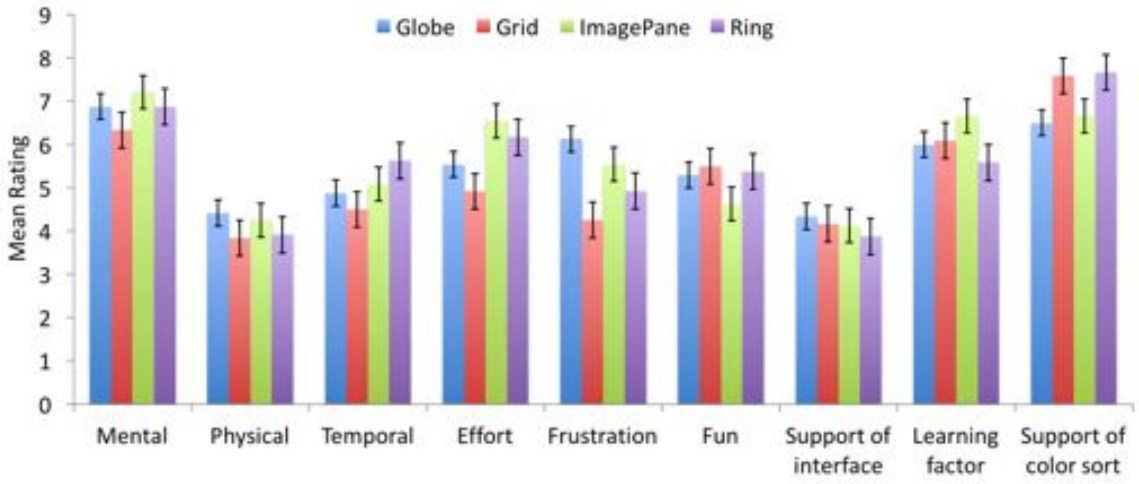

Figure 5.7: Mean rating of questionnaires (lower is better but for *Fun*, *Support of interface* and *Support of color sort* (Error bars: +/- SE).

5.11 Conclusions - Tablet and Smartphones Studies

In the tablet user study of chapter 4 a trend was already visible that the Ring and Globe in fact perform very similarly. This trend continued in the smartphone user study and was also supported by the rather strong performance of the Grid interface. As noted before, users might need more training to be able to use the features of the 3D interfaces (especially the ring) to their maximum. In case of the Grid, participants already knew how to use it from their experiences with their own smartphones. Therefore, the learning curve was rather flat. Furthermore, they were not mentally burdened to learn a new interaction technique and could focus completely on solving their task.

At the same time, the quite equal geometric mean search times for all target groups with the ImagePane have to be considered. It seems to have the ability to produce good results regardless of where the target image is placed or which dominant color it has. It may also utilizes the advantages of the color sorting to the possible maximum since it exposes the sorting completely to the user right at the start without any need for interaction.

It also became clear that user simulations and real user studies can produce quite different results. When only looking at the results of the simulation, the ImagePane would be by far the best performing smartphone image browsing interface. The following user study could not confirm this result at all. The problem with user simulations is that they assume an optimal environment that can rarely be realized in reality. Also, not all users are equal and have perfect understanding of the interface and color sorting. In the user studies it was often noticed that participants had difficulties for deciding on the dominant color in an image. As a consequence, they looked at the wrong area and wasted precious time.

An area where simulation and user study agreed was the linear increase of search time and interaction effort in case of the Grid interface. The lower the target image was placed in the Grid the longer it took participants to find it. This was not at all the case with the other interfaces.

An additional idea that was discovered during the user studies concerns a difficulty rating for images. In the data there are strong deviations in search time between images of the same target group, although they were located next to each other. This was true for a specific set of images across multiple participants. It seems that images differ in terms of the search difficulty depending on the content itself and by what kind of images they are surrounded with. It could be advantageous to do some work on how to automatically calculate a difficulty rating for images in such scenarios. Such a difficulty rating could be used in future KIS-like test setups and give interesting insights in the way interfaces perform on different image difficulty levels.

CHAPTER

6 Mobile Video Browsing with the Keyframe-Navigation-Tree

As discovered in chapter 3 users typically have only a low number of videos on their smartphones and tablets. Nevertheless, the need to find important scenes inside a single video still exists. Especially long videos can be a problem. Therefore, in this chapter the Keyframe-Navigation-Tree Video Browser (KNT-Browser) is introduced.[1] It uses a novel approach of (sub-)shot detection and shot visualization to help users navigate inside videos faster and more efficient.

6.1 Sub-Shot Detection

In contrast to traditional shot detection approaches the KNT-Browser utilizes a method that works on a sub-shot level [LXSS14]. This has the advantage that it also works for videos with long shots, e.g. extensive camera pans over a landscape.

Typically, keyframe selection approaches are based on shot boundary detection. A shot is generally a group of continuous video frames with consistent visual characteristics such as color, texture, and motion, captured from a single camera at a time. The "gap" between

[1] Please note that this chapter is adapted from [HSX15a]

two neighboring shots is called a *shot boundary*. Furthermore, the transitions between shots can be classified into two rough groups: *hard cuts* and *gradual transitions*. On one hand, a hard cut is an abrupt shot change that occurs between two continuous frames. Gradual transitions on the other hand occur over multiple frames, like fade-in/out, dissolve and wipe animations, as described by Hanjalic [Han02] and Lienhart [LPE97].

To handle video sequences of any kind, a very general information theoretic measure, the *Jensen-Shannon Divergence* (*JSD*) [XPCL$^+$10], for computing the difference between video frames is used. As a matter of fact, any general metric (such as the widely used f-divergences [LXSS14]) that is computed efficiently, can be used for this purpose. As for the computational mechanism of the keyframe selection, a simple and effective shot-based approach is utilized. In this approach, a video sequence is divided into non-overlapping shots. Considering the possibilities of gradual transitions between neighboring shots, a shot is then possibly divided into sub-shots. Finally, one frame is chosen for each (sub-)shot. Here $D(f_i||f_{i+1})$ is used to represent the difference between the i and $i+1$ video frames, calculated based on their corresponding normalized intensity histogram distributions, f_i and f_{i+1}. In practice, the distance between video frames is the sum of correspondences for the three RGB channels.

The computing procedure for the keyframe selection can be described as follows (further details can be found in [LXSS14, XLY$^+$12, XLL$^+$14, XPCL$^+$10]). The Ds between each pair of two consecutive video frames are obtained first. In order to locate D spikes indicating the existence of shot boundaries, a ratio $\delta = \frac{D}{D_w}$, where D_w is a local average of D on a w size temporal window, is used. A shot boundary is located when δ is greater than a pre-defined number δ^*. Next, a shot is grouped into several sub-shots if the content change of this shot is significant enough. The gradient of D_w, calculated as $\Delta(j) = D_w(f_j||f_{j+1}) - D_w(f_{j-1}||f_j)$, is employed to detect a significant content change. To prevent the use of small outliers of Δ, the local average of Δ on a temporal window, denoted as Δ_w, is actually used for this sake. Within a shot, if $|\Delta_w(j)|$ is greater than a pre-determined number Δ_w^*, then a significant content change inside this shot appears around the frame f_j. The left and right closest frames to f_j, f_m and f_n that satisfy $\Delta_w(f_m) \simeq 0$ and $\Delta_w(f_n) \simeq 0$, are respectively

located as the beginning and end of the significant content change. In fact, f_m and f_n are respectively the left and right boundaries of a sub-shot $[f_m, f_n]$ with the significant content change for this shot. In this way, a shot is segmented into several sub-shots according to the boundaries of all the sub-shots with significant content changes. For a sub-shot with a significant content change, the frame being most similar to all the others is selected as a representative keyframe. For a sub-shot without significant content change, the center frame of it is used for this purpose.

6.2 Interface Design

The design of the interface that can be seen in Figure 6.1 is based around the idea of *frame stripes* as proposed by Schoeffmann et al. [STB10] (also known as *MO-images* introduced by Mueller-Seelich and Tan [MST00]). Frame stripes are generated by taking the center pixel column of every frame and adjacently visualizing them in a horizontal stripe-like manner. They give users an idea about the structure (and to some degree about the content) of a video while being very compact.

The idea is extended in the KNT-Browser. Instead of visualizing every frame, only the representative frame of a sub-shot is visualized. The resulting stripe will therefore be called shot-stripe. Moreover, instead of using only a slice of one pixel width, the slices are made wider to give users a better idea about the content of the sub-shot.

Furthermore, the KNT-Browser offers not one but three shot-stripes. Each of the shot-stripes operates with another slice width. This enables users to switch seamlessly between different visualizations depending on their current needs. The different shot-stripes are defined *small*, *medium* and *full*.

Figure 6.1: Interface of the KNT-Browser. **Top:** preview player window. **Bottom:** three shot-stripes with different level of detail.

The *small* shot-stripe uses 20% of the frames' original width. It is meant to give a first insight into the content of a video while providing good overview.

The *medium* shot-stripe shows about one third of the frames' actual width. It is a compromise between presenting more content and slightly increased space requirements. Users can fall back to this type if they are not able to find what they are looking for by using the *small* shot-stripe.

Finally, the *full* shot-stripe visualizes the entirety of the frames. It offers the best view of the sub-shots content but requires considerably more space. This stripe should be used for very detailed inspection.

In addition to the shot-stripes, the KNT-Browsers interface also provides a big preview player. It is meant to be used for very thorough inspection of the content inside a sub-shot. All interface controls (preview player and shot-stripes) are connected and in sync to each other. That means when users operate with one of them, the changes are automatically reflected in the other controls. When users interact with a stripe (scrolling or selecting) the other stripes and the player reposition their video position accordingly. This is visualized with a red line that crosses all three shot-stripes. It is meant to represent the concept of a VCR-like playhead (see Figure 6.2).

The idea of browsing at different granularity levels is inspired by works shown by Cobarzan and Schoeffmann [CHDF14, SC13]. They investigated users' navigation behavior with common video players and discovered that for KIS tasks users typically navigate in a *coarse-to-fine* grained manner. First, users try to narrow down the search to a specific temporal area in the video. They then continue by switching to a closer inspection of that predetermined area. The KNT-Browser supports this natural search behavior.

Figure 6.2: The three shot-stripes of the KNT-Browser in greater detail.

6.3 Evaluation

For the evaluation the KNT-Browser was implemented using an Apple iPad Air with iOS 7 and Objective-C as programming language. To be able to compare it to the standard interface also a standard video player was implemented, like it is typically preinstalled on those devices. It features a seeker-bar as well as a play/pause button and a label with the current timecode (see Figure 6.3.1). It is identical to the default video player that is preinstalled on iOS 7 devices.

6.3.1 User Study

The user study was designed around the notion of KIS-like tasks in videos. As dataset and search tasks the data from the *single run* session of the Video Browser Showdown competition (VBS) in 2014 [SB12] is used. It is structured in ten different search tasks. Each task has to be performed with one out of ten videos. Moreover, each video has an approximate duration of one hour.

In total 20 participants (five females) took part in the user study, aged from 18 to 40 (mean 28.15 years, s.d. 6.08). Every participant performed five tasks with the default video player interface and five tasks with the KNT-Browser. To avoid learning effects a latin-square principle with random order is used.

Figure 6.3: The default video player interfaces that was used for the evaluation.

The study process was inspired from the last years VBS: for each task a 20 seconds target clip is displayed on the screen, which the participants then have to find in a longer video. Participants have a three minute time window available to complete each task. In the case they miss the deadline the trial is aborted and marked as *timed out* in the log data. In contrast to the VBS no kind of scoring was used that is visible to the user. It was decided to avoid giving them feedback about their performance to prevent influencing their search behavior in any way. Instead, the search time and trial success are logged in the background. In accordance with the VBS a tolerance of ±5 seconds at the beginning and the end of the target segment is granted. A test session typically lasted for about half an hour per participant.

After users had finished five trials with one interface, they had to fill out a questionnaire with Likert-scale ratings about the subjectively perceived workload of the interface, according to the NASA Task-Load-Index (TLX) [HS88]. They then continued with five new trials with the seconds interface.

6.3.2 Analysis of Trial Times

Only successful trials were used for the analysis of trial times, e.g., trials where participants found the right video sequence in time. In a first step outliers were removed from the gathered data (± 2 s.d.). The remaining trial times were positively skewed for both interfaces. Therefore, a logarithmic transformation was performed to achieve a close to normal data distribution before the data was analyzed. Figure 6.4 (left) shows geometric means for the two tested interfaces (i.e., the antilog of the mean of the log-transformed data): **28.11** seconds for the KNT-Browser and **44.18** seconds for the default video player. A *dependent paired-samples t-test* showed that the difference between the interfaces is statistically significant ($t(19) = -3.937, p < 0.005$). Furthermore, the variance of search time was much smaller for the KNT-Browser than for the default video player, which indicates that the performance of the KNT-Browser is less dependent on the type of content.

When comparing these results to the average task solve time achieved by sophisticated video browsing tools in the VBS 2014 competition (**20.58** seconds in the *Visual KIS/Experts* run [Sch14]), we can see that the achieved performance in this user study (mainly with

novices) is quite remarkable. With the KNT-Browser the users were even faster than the experts in the VBS 2013 competition (also with sophisticated tools) [SAB+14], who required **40.5** seconds on average to solve KIS tasks in different but similarly long videos. For the same data set novices in a baseline study with a common video player on a desktop PC required **57.9** seconds on average [SAB+14].

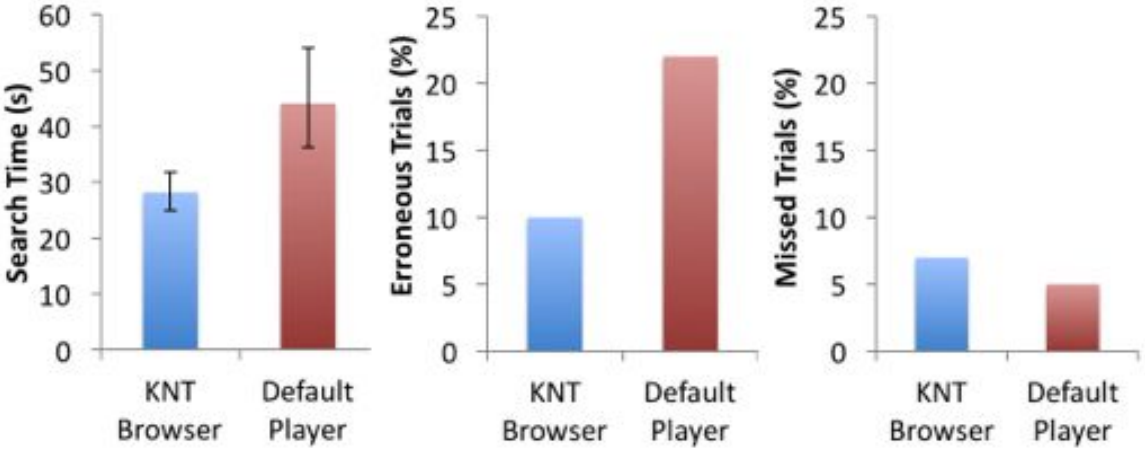

Figure 6.4: KNT-Browser vs. default player - **Left:** geometric mean trial times (error bars: 95% confidence interval) **Middle:** erroneous trials **Right:** timed out trials.

6.3.3 Errors and Timeouts

Also the number of errors were investigated that occurred with each interface. From a total of 100 search tasks performed with each interface (each of the 20 users solved 5 tasks with one interface), **22** were not correctly solved with the default player, whereas with the KNT-Browser only **ten** tasks were erroneous (see Figure 6.4 middle). A closer inspection revealed that almost half of the erroneous trials were caused by one specific task, which required finding a segment that reappeared in similar form in other locations of the corresponding video.

The number of timeouts (i.e., where a user needed more than three minutes for a task) was quite balanced between both interfaces. From a total of 200 trials with both interfaces, only **five** could not be solved with the default player, whereas **seven** could not be solved

with the KNT-Browser (Figure 6.4 right). The timeouts were clearly task-dependent, as all 12 timeouts were almost uniformly caused by three specific tasks.

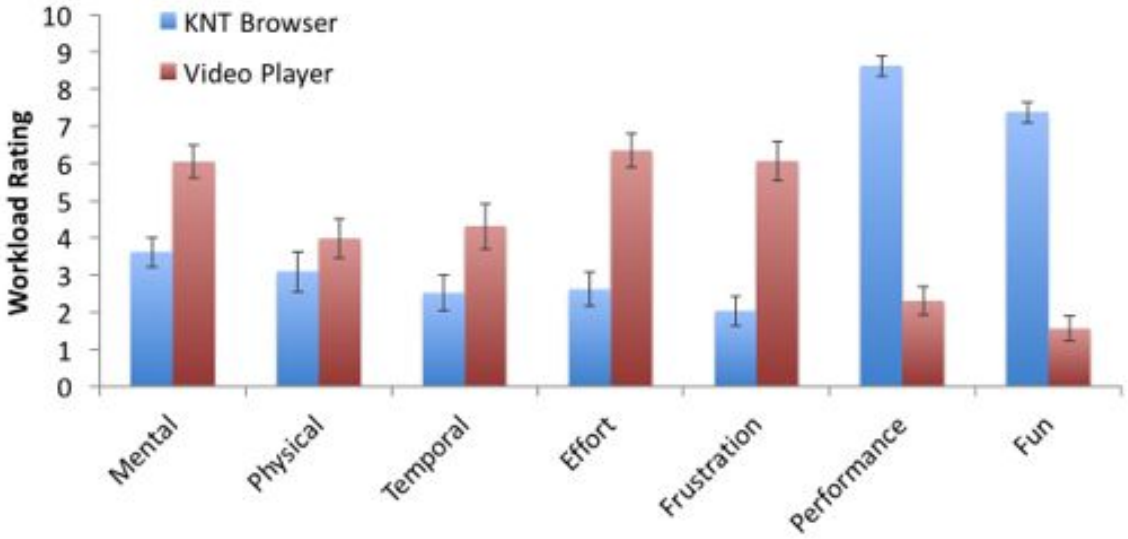

Figure 6.5: Perceived workload rating (error bars: ±s.e. of the mean)

6.3.4 Subjective Rating

Statistical analysis (Wilcoxon signed-rank tests) of the subjective evaluations of the interfaces showed that the KNT-Browser performed significantly better in all seven categories of the NASA Test-Load-Index [HS88], as shown in Figure 6.5. More specifically, all study participants found that the KNT-Browser better supports KIS-like tasks in videos ("Performance" in the figure, $Z = -3.920,\ p < 0.0005$), that is less frustrating than the default player ($Z = -3.921,\ p < 0.0005$), and that it is more fun to use than the default player ($Z = -3.920,\ p < 0.0005$). Furthermore, the vast majority found that the KNT-Browser requires less effort ($Z = -3.659,\ p < 0.0005$), less mental demand ($Z = -2.576,\ p = 0.01$), less physical demand ($Z = -2.535,\ p = 0.011$), and produces less temporal pressure ($Z = -2.240,\ p = 0.025$) than the default player.

6.3.5 Preferred Interface

In the questionnaires at the end of a test session the participants were asked about their preferred interface. The vast majority, more precisely 17 out of 20 users (85%), voted for the KNT Browsing interface, whereas only three preferred the Video Player.

6.4 Summary

As the results of the evaluation showed, the KNT-Browser with its approach to navigation interaction and shot detection, can improve the search experience considerably. It provides a good insight and overview of the contents of a single video. At the same time the three shot-stripes with different granularity levels enable users to also examine shots and sub-shots in great detail.

Moreover it utilizes a novel segmentation method on a sub-shot level. The method has the advantage that it is usable for several different domains. It works with videos containing very long shots (recordings of landscapes) as well as videos with very short or only a single shot (medical domain). Regardless of the domain it can deliver good content summarization.

The KNT-Browser succeeded in quantitative measures (search time, errors) as well as in qualitative regards (workload inquiries). The majority of study participants (85%) also stated that they preferred it over the default video player. The insights of its evaluation influence the design of new video browsing tools. At the time this thesis is written two of them are in prototype stages but not yet evaluated, which is reserved for future work.

CHAPTER

7 Discovering Limits: Mobile OpenCV Performance

During the development of the before presented image and video browsing concepts, the design process always involved balancing the processing demands with available features. As it was reported, today's smartphones and tablets indeed can offer great performance, but especially processing needs of complex content analysis methods can be a problem. That is the reason why content-related processing in the earlier presented interfaces was done offline.

Nevertheless, the idea of on-the-fly and mobile analysis of content directly on the devices is still very attractive. However, it is essential to be conscious about which methods are feasible in a mobile environment and to carefully consider the relation of cost and benefit. Otherwise, user interaction and experience can seriously suffer and render the whole approach useless in the real world.

As a consequence two performance evaluations were performed that focused specifically on the given problem. For this, a popular set of functions and algorithms provided by the OpenCV library were tested with a collection of popular smartphones and tablets. OpenCV is a freely available open source library for computer vision applications and it is widely used for content analysis purposes. Implementations are available for Windows, Linux, Mac OS X as well as iOS and Android.

The given figures should be interesting for both, research and product development. In the current literature there exists a lack of such evaluation results on which both fields can base their considerations for future research and implementations.

In this chapter the results of the more recent study are reported, as it includes a similar setup of test cases and covers iOS as well as Android devices[1]. As we have seen in chapter 3, these two are the dominant mobile platforms. For further reference, the results of the first performance evaluations can be found in Appendix A.

7.1 Android Vs. iOS

The following set of measurements include a broad range of Android and iOS devices in direct comparison. In addition to measuring the execution time of OpenCV functions, battery usage is also investigated. It is included in response to feedback that was received regarding earlier OpenCV experiments. Battery performance is of great importance to any type of mobile device. Moreover, keypoint detection and descriptor extraction is measured separately. Furthermore, the measurements focus on single core execution performance.

7.1.1 Setup

The experiment is grouped into two phases with the following setup:

- **Phase 1:** grayscaling, image blurring, face and edge detection;
- **Phase 2:** keypoint detection and descriptor extraction.

For both phases a set of 250 images[2] was prepared, drawn from the INRIA Holiday dataset [JDS08]. All images have a resolution of about three mega pixel but differ in their orientation (portrait vs. landscape). Current and future computer vision applications will use heterogeneous data from multiple sources, including media captured directly with the

[1]Please note that the content of this chapter is adapted from [CHSP15]

[2]The complete file listing can be downloaded at `http://hudelist.org/research/phdthesis/250InriaListing.txt`

devices as well as media coming from other sources. Therefore, it makes sense to introduce slight variations in the dataset. On average the images have a resolution of 2547 vertical and 2016 horizontal pixels. Furthermore, in order to cover the variety of image and video capture capabilities of older and newer devices three subsets are created:

- **3MP:** all 250 images in their original size of about three mega pixel (avg.: 2456x2107).
- **3MP50:** all 250 images reduced to 50% in their original resolution (avg.: 1228x1053).
- **3MP25:** all 250 images reduced to 25% of their original resolution (avg.: 614x526).

The rest of the experiment setup is similar to the earlier study. In phase one and two function calls are measured with each image of a dataset five times in a row. The measurement times are averaged in order to even out possible interventions by the OS. The averaged times for each image are again averaged to get an overall performance measure for each specific function.

The devices that were used for the experiment included the (at that time) flagship Android and iOS devices. The latest Nexus and Galaxy product lines were included as well as different generations of Apple's iPads and iPhones. A short overview of the device specifications can be found in Table 7.1. The CPUs on the iPad Air and the iPhone 5S have a 64-bit architecture while all the other devices have a 32-bit CPU architecture. The System-On-Chip (SoC) information is presented in Table 7.2. All Android devices use Android version 4.4.2 except for the GalaxyNote 10.1, which uses version 4.3. All iOS devices use iOS version 7.1.2. Moreover, on all devices OpenCV release 2.4.9 was installed.

7.1.2 Results

In the following, completion time as well as battery usage are reported. After that, the results are discussed. The keypoint detection and descriptor extraction operations are measured independently. The battery usage measurements were performed with 1% granularity steps for the Android devices and with 5% granularity steps for iOS devices. The steps are higher for iOS because Apple's API does not provide more detailed information. This means that

Device	CPU GHz/cores	Memory	Screen	Resolution	Battery
Galaxy Note 10.1 (2014)	1.95/8	3 GB	10.1"	2560×1600	8220 mAh
Galaxy Note 3	2.3/4	3 GB	5.7"	1920×1080	3200 mAh
Galaxy S4	1.9/4	2 GB	5.0"	1920×1080	2600 mAh
Nexus 7 (2013)	1.5/4	2 GB	7.0"	1920×1200	3950 mAh
Nexus 7	1.2/4	1 GB	7.0"	1280×800	4325 mAh
Nexus 5	2.3/4	2 GB	4.95"	1920×1080	2300 mAh
iPad Air	1.4/2	1 GB	9.7"	2048×1536	8820 mAh
iPad 4	1.4/2	1 GB	9.7"	2048×1536	11560 mAh
iPad 3	1/2	1 GB	9.7"	2048×1536	11560 mAh
iPad Mini 1	1/2	512 MB	7.9"	1024×768	4382 mAh
iPad 2	1/2	512 MB	9.7"	1024×768	6930 mAh
iPhone 5S	1.3/2	1 GB	4.0"	1136×640	1560 mAh
iPhone 5	1.3/2	1 GB	4.0"	1136×640	1440 mAh
iPhone 4s	0.8/2	512 MB	3.5"	960×640	1432 mAh

Table 7.1: Android and iOS devices specification breakdown.

Device	CPU SoC
Galaxy Note 10.1 (2014)	Samsung Exynos 5420
Galaxy Note 3	Qualcomm Snapdragon 800
Galaxy S4	Qualcomm Snapdragon 600
Nexus 7 (2013)	Qualcomm Snapdragon S4 Pro
Nexus 7	Nvidia Tegra 3 T30L
Nexus 5	Qualcomm Snapdragon 800
iPad Air	Apple A7
iPad 4	Apple A6X
iPad 3	Apple A5X
iPad Mini 1	Apple A5 (2nd Gen.)
iPad 2	Apple A5
iPhone 5S	Apple A7+M7
iPhone 5	Apple A6

Table 7.2: Android and iOS System on Chip information.

there is a ±1% error margin for the Android devices and a ±5% error margin for the iOS devices since it is possible that a previous test has ended just under the threshold and part of it's battery consumption gets counted for the following test. The tests were started on all devices with the battery fully charged. When the battery was completely depleted (the device turned itself off) it was recharged to 100%. After that, the device continued by re-running the test at which it had turned off.

Common OpenCV Operations The same common OpenCV operations of the earlier iOS-centric measurements are evaluated: grayscaling, blurring, face detection, RGB and HSV histograms and edge detection. Also, the same configuration for each of the functions was used. The result values are presented in Table 7.3 and Table 7.4 for the 3MP and 3MP50 datasets.

Blurring is performed using the *GaussianBlur(...)* function with a *kernel size* of *21×21* and *sigma* set to *8.0* in order to produce still recognizable result images (3MP avg. resolution: 2456x2107, 3MP50 avg. resolution: 1228x1053, 3MP25 avg. resolution: 614x526). For face detection a trained *CascadeClassifier* is used and its *detectMultiScale(...)* function. Before the actual detection, the images are grayscaled. The grayscaling itself is not included in the time measurement. The other used parameters are set as follows: *scaleFactor* set to 1.1, *minNeighbors* set to 2 and *minimum size* set to 30×30.

The RGB histograms that are created use 256 bins with *uniform* set to *true* and *accumulate* set to *false*. The range of each bin is between 0 and 255. Also, the HSV histograms generate 30 hue levels and 32 saturation levels with default ranges. Furthermore, *uniformity* is set to *true* and *accumulation* is set to *false*.

To detect edges with Canny the images are first grayscaled and then blurred. For blurring, a *kernel size* of 5×5 and a *sigma* of 1.2 is used (both operations are not included in the actual time measurement).The *Canny(...)*-function is measured with thresholds one and two set to 0 and 50 respectively.

For a visual aid when analyzing the data in Table 7.3 and Table 7.4, please refer to Figure 7.1 and Figure 7.2. Both use a logarithmic scale for representing the measured

Device	Grayscale	Gaussian Blur	Face Detection	RGB Histogram	HSV Histogram	Canny Edge Detection
Galaxy Note 10.1	8.35	592.62	2471.53	11.64	6.00	145.01
Galaxy Note 3	10.05	680.84	3297.80	19.33	6.40	153.25
Galaxy S4	14.55	886.78	4261.88	28.08	9.96	205.38
Nexus 7 (2013)	14.60	1014.20	3439.15	30.03	9.22	240.46
Nexus 7	11.43	1501.77	3273.17	27.55	20.56	193.18
Nexus 5	7.94	840.87	3494.73	25.19	7.65	144.22
iPad Air	12.95	1260.53	7930.91	69.84	25.06	282.01
iPad 4	21.28	2464.97	10400.87	61.26	33.38	384.51
iPad 3	51.97	4604.35	16556.63	81.00	87.37	797.74
iPad Mini 1	31.26	2769.81	9954.92	47.69	52.92	493.89
iPad Mini 2	13.62	1368.43	8771.00	79.85	27.76	328.01
iPad 2	30.51	2868.00	9842.43	47.09	52.20	488.23
iPhone 5S	8.12	1106.79	6338.24	42.84	15.98	179.60
iPhone 5	13.27	1705.18	6615.14	39.27	21.05	245.03

Table 7.3: Common Operations (values in ms) - 3MP images.

values of common operations in the case of the 3MP and 3MP50 datasets. Interestingly, no clear winner can be determined between Android and iOS. Furthermore, in some cases unexpected results are recorded, as for example the iPhone 5 beats its successor (iPhone 5S) in RGB histogram generation, or the iPhone 5S beating the iPad Air, which uses the same processor but with slightly increased clock rate.

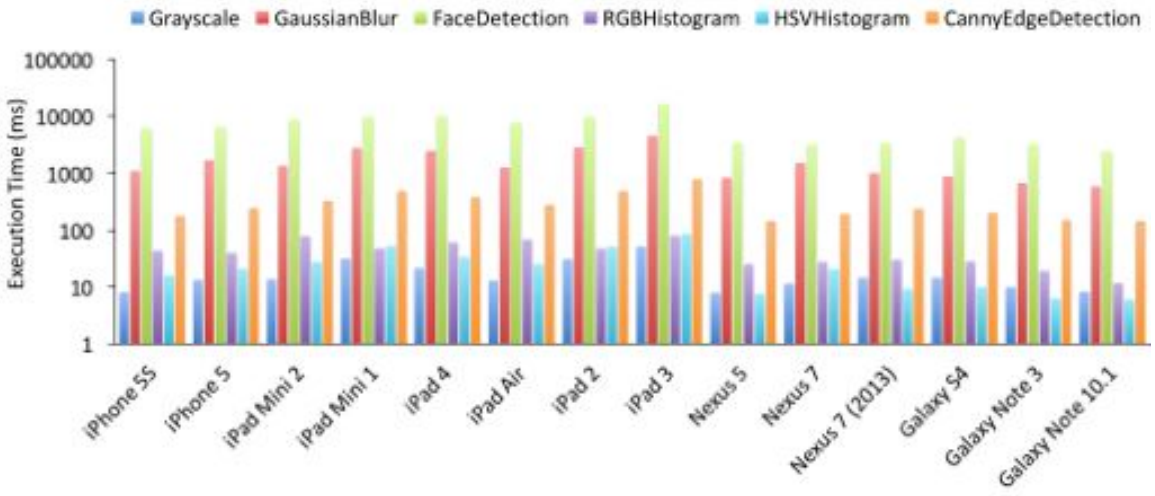

Figure 7.1: Measured results of common operations in the case of the 3MP dataset.

Device	Grayscale	Gaussian Blur	Face Detection	RGB Histogram	HSV Histogram	Canny Edge Detection
Galaxy Note 10.1	2.36	135.85	556.21	4.08	1.82	37.20
Galaxy Note 3	3.33	151.62	734.67	7.90	2.80	44.07
Galaxy S4	3.99	181.88	933.57	9.89	4.06	61.18
Nexus 7 (2013)	4.63	213.22	780.29	8.84	4.30	61.81
Nexus 7	3.33	358.98	798.09	7.11	5.36	46.60
Nexus 5	2.55	220.05	867.93	8.04	3.17	65.85
iPad Air	3.32	270.05	1815.28	19.26	6.17	69.23
iPad 4	6.20	534.86	2666.15	16.02	9.13	98.01
iPad 3	13.35	1077.73	3936.46	20.48	21.95	196.59
iPad Mini 1	13.23	1074.42	3971.17	19.84	21.88	199.14
iPad Mini 2	3.43	289.37	2376.85	22.43	6.88	79.66
iPad 2	12.84	1076.47	3930.68	19.64	21.59	196.88
iPhone 5S	3.43	288.90	2858.57	16.82	6.60	73.12
iPhone 5	6.66	573.81	2951.05	16.91	9.60	102.293

Table 7.4: Common Operations (values in ms) - 3MP50 images

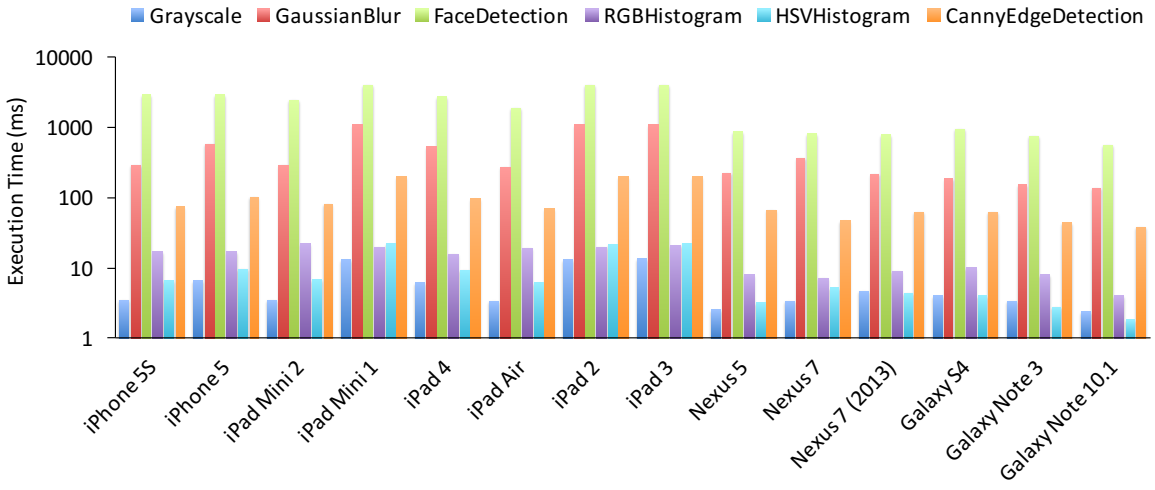

Figure 7.2: Measured results of common operations in the case of the 3MP50 dataset.

As it can be seen in the results, the most demanding functions are face detection, Gaussian blur and Canny edge detection, while the least demanding are HSV histogram generation and grayscaling. The corresponding battery drop levels for completing the considered test sequences of each function for the 3MP and 3MP50 datasets are presented in Table 7.5. As expected, there is a strong correlation between battery consumption and the time needed

Device	Grayscale		Gaussian Blur		Face Detection		RGB Histogram		HSV Histogram		Canny Edge Detection	
	D1	D2	D1	D2	D1	D2	D1	D2	D1	D2	D1	D2
Galaxy Note 10.1	1	0	2	2	40	9	1	0	0	0	3	1
Galaxy Note 3	0	0	13	4	63	14	2	0	1	1	5	1
Galaxy S4	1	0	17	3	86	19	3	0	0	1	5	1
Nexus 7 (2013)	0	0	13	2	86	20	2	0	1	0	5	1
Nexus 7	0	0	17	0	83	18	3	1	2	0	5	1
Nexus 5	1	0	15	5	89	21	2	1	1	0	4	1
iPad Air	0	0	10	0	65	15	0	0	0	0	5	0
iPad 4	0	0	20	5	75	20	5	0	0	0	5	0
iPad 3	5	0	30	5	>100	35	5	0	0	0	10	0
iPad Mini 1	5	0	20	5	80	40	0	0	5	0	5	0
iPad Mini 2	0	0	10	5	70	15	0	0	0	0	5	0
iPad 2	0	0	20	5	75	35	0	5	0	0	5	0
iPhone 5S	0	0	10	5	90	35	0	5	0	0	5	0
iPhone 5	0	0	15	10	>100	45	0	0	5	0	5	0

Table 7.5: Battery demand (in %) for operations in the common group after completing the test sequence of 250 images from the 3MP (D1) and 3MP50 (D2) datasets. Entries of ">100" indicate that a single full charge was not sufficient to complete the test sequence.

to complete the tests. The largest battery drain can be observed for face detection. This is true for all devices. Moreover, in the case of the iPad 3 and the iPhone 5 a full battery charge is not sufficient for completing the face detection test for all the images in the 3MP dataset.

Keypoint Detection/Descriptor Extraction Next, keypoint detection and descriptor extraction was examined. This time, detection and extraction were measured separately in accordance with response to feedback that was received to the first performance measurements. The following algorithms are included: ORB, BRIEF, BRISK, SIFT, SURF, FREAK and FAST. In the case of BRIEF and FREAK GoodFeaturesToTrack was used for keypoint detection followed by descriptor extraction.

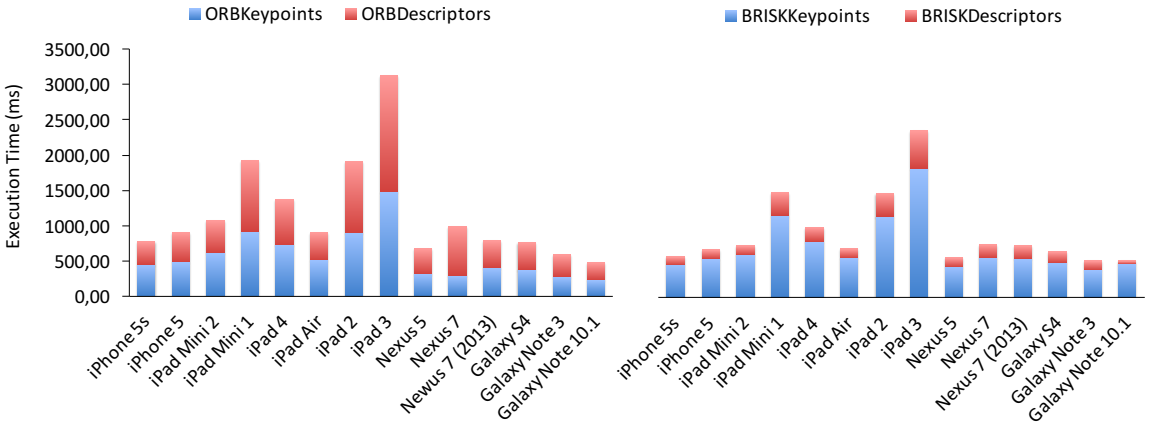

Figure 7.3: Execution times of ORB (left) and BRISK (right) keypoint detection and descriptor extraction.

In Figures 7.3 and 7.4 the measurement values are visualized corresponding to keypoint extraction as well as descriptor computation for ORB, BRISK, FREAK and FAST for the 3MP dataset. Again, the rather bad performance of the iPad 3 in comparison with its predecessor, the iPad 2 is visible.

Within the Android devices, the keypoint detection in case of ORB clearly outperforms the iOS devices. BRISK shows similar results across all devices with the exception of the iOS devices mentioned earlier. The same is true in the case of FREAK with the observation that the Galaxy Note 10.1 and Nexus 5 provided the fastest results. Moreover, keypoint detection with FAST required less time on all the Android devices.

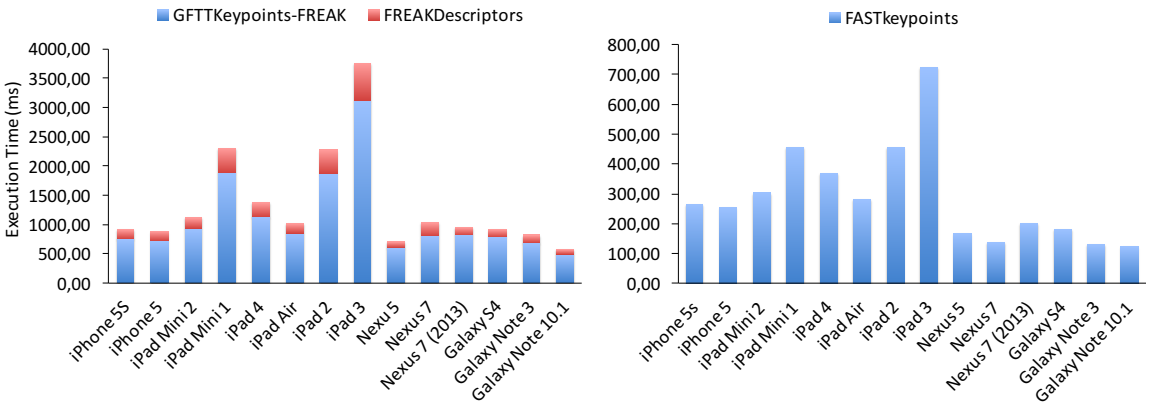

Figure 7.4: Execution times of FREAK (left, keypoint detection & descriptor extraction) and FAST (right, keypoint detection).

In the following the results regarding SIFT, SURF and BRIEF are discussed. The measurements in those cases also include the time needed to detect keypoints as well as the time needed to extract the corresponding descriptors. It is important to note that due to patent issues, SIFT and SURF are not included in the official release package of OpenCV for the Android platform. They are relocated into the *nonfree* module of the library. In order to perform the measurements it is necessary to make a build of this *nonfree* module for Android native projects. One option is to rebuild the whole OpenCV library. For this experiment however, only the missing *nonfree* module[3] was rebuilt (using the Android NDK, Revision 9c - December 2013).

The exact measured values for the 3MP, 3MP50 and 3MP25 datasets are listed in Table 7.6, Table 7.7 and Table 7.8. Although in the case of BRIEF the Android devices achieve better results, in case of SIFT and SURF the measured values are considerably worse than in comparison to the iOS devices.

In case of the 3MP dataset, the SIFT measurements could not be performed on any of the devices because the test applications crashed on both Android and iOS with an insufficient memory error message (therefore they are omitted from Table 7.6). This also happened in

[3]Following the instructions at `http://tinyurl.com/oblle6l`

the case of the 3MP50 dataset for the iPad Mini 1 and iPad2. The corresponding values are marked as '−' in Table 7.7 and Table 7.9, which shows battery drain results in the case of BRIEF, SIFT and SURF for the three considered datasets. Please note that for the 3MP dataset, the SURF measurements needed more than one full charge to complete on all tested devices.

Device	BRIEF		SURF	
	Keypoints	Descriptors	Keypoints	Descriptors
Galaxy Note 10.1	488.78	42.97	82662.90	184723
Galaxy Note 3	677.24	52.20	58601.10	134267.00
Galaxy S4	801.14	75.76	74455.90	173836
Nexus 7 (2013)	824.41	73.78	58248.60	136172.00
Nexus 7	811.50	71.41	45270.00	97972.20
Nexus 5	582.38	48.87	63328.40	148495.00
iPad Air	901.05	54.38	7115.26	20084.18
iPad 4	1134.90	79.62	6552.49	17761.25
iPad 3	3116.00	177.17	12447.22	35201.64
iPad Mini 1	1879.90	126.21	8810.79	22985.91
iPad Mini 2	952.29	56.97	7735.29	21879.85
iPad 2	1866.40	123.42	8586.47	22650.45
iPhone 5S	569.57	36.58	6631.87	18658.79
iPhone 5	729.66	56.50	5120.15	13744.48

Table 7.6: Average execution times (in ms) for keypoint detection and descriptor extraction when testing 250 images of the 3MP dataset.

Device	BRIEF		SIFT		SURF	
	Keypoints	Descriptors	Keypoints	Descriptors	Keypoints	Descriptors
Galaxy Note 10.1	126.26	16.82	18553.60	20168.90	24220.60	51071.20
Galaxy Note 3	157.51	21.18	21520.80	27542.70	15096.30	33957.30
Galaxy S4	211.50	30.13	26907.70	33548.60	20023.70	46063.10
Nexus 7 (2013)	225.88	33.10	28082.10	33697.40	15718.30	34687.50
Nexus 7	220.16	33.97	38399.30	47780.40	12494.10	27438.70
Nexus 5	180.35	25.91	27057.40	34179.10	18131.50	41822.00
iPad Air	226.52	16.49	3276.46	4780.03	1890.82	5106.70
iPad 4	286.16	25.47	4883.35	6847.57	1846.23	4902.68
iPad 3	756.95	61.63	11560.26	15891.07	3449.07	9811.72
iPad Mini 1	756.18	62.07	–	–	3492.48	9879.97
iPad Mini 2	240.21	17.35	3548.15	5186.26	2236.38	6059.69
iPad 2	751.41	60.96	–	–	3436.40	9795.54
iPhone 5S	218.44	15.75	4209.18	6161.39	2670.63	7242.44
iPhone 5	304.46	26.66	5191.81	7302.17	2085.50	5567.79

Table 7.7: Average execution times (in ms) for keypoint detection and descriptor extraction when testing 250 images of the 3MP50 dataset.

Device	BRIEF		SIFT		SURF	
	Keypoints	Descriptors	Keypoints	Descriptors	Keypoints	Descriptors
Galaxy Note 10.1	33.59	7.89	4833.04	5371.13	6201.35	12579.20
Galaxy Note 3	32.55	8.00	4757.97	6533.47	4156.58	8940.18
Galaxy S4	47.74	11.77	6283.44	8221.01	5253.89	11394.90
Nexus 7 (2013)	50.72	12.12	7233.23	9301.98	4043.44	8614.13
Nexus 7	50.33	16.41	9445.54	12471.60	3250.60	6820.61
Nexus 5	39.55	9.75	5940.75	8033.79	4888.02	10580.00
iPad Air	58.52	5.76	914.51	1407.81	587.21	1447.07
iPad 4	76.20	9.38	1185.85	1767.18	564.72	1360.12
iPad 3	187.33	26.09	2874.33	4191.65	905.31	2406.50
iPad Mini 1	186.45	26.56	2955.26	4310.72	919.02	2428.89
iPad Mini 2	61.80	6.09	956.89	1475.02	623.83	1540.25
iPad 2	186.10	25.96	2888.90	4224.87	899.57	2397.98
iPhone 5S	55.33	5.43	1127.66	1742.89	677.56	1674.53
iPhone 5	80.94	9.82	1300.93	1933.76	514.59	1267.40

Table 7.8: Average execution times (in ms) for keypoint detection and descriptor extraction when testing 250 images of the 3MP25 dataset.

Device	BRIEF			SIFT			SURF		
	D1	D2	D3	D1	D2	D3	D1	D2	D3
Galaxy Note 10.1	9	2	1	–	55	15	>100	>100	30
Galaxy Note 3	15	3	1	–	93	19	>100	98	24
Galaxy S4	18	5	2	–	>100	25	>100	>100	33
Nexus 7 (2013)	15	4	1	–	94	18	>100	>100	33
Nexus 7	20	6	1	–	>100	35	>100	>100	26
Nexus 5	15	4	1	–	>100	25	>100	>100	34
iPad Air	5	5	0	–	60	15	>100	55	15
iPad 4	10	5	0	–	80	20	>100	45	15
iPad 3	25	10	0	–	>100	45	>100	100	25
iPad Mini 1	10	5	5	–	–	55	>100	>100	25
iPad Mini 2	10	0	0	–	70	15	>100	60	15
iPad 2	15	5	0	–	–	50	>100	100	25
iPhone 5S	10	5	0	–	>100	35	>100	>100	35
iPhone 5	10	5	0	–	25	40	>100	>100	30

Table 7.9: Battery drop levels (values in %) for the 3MP (D1), 3MP50 (D2) and 3MP25 (D3) datasets for keypoint detection and descriptor extraction.

7.1.3 Discussion

In general Android devices showed better performance in this experiment setup. The reasons for this are not completely clear. One reason could be that this difference is mainly caused by more powerful hardware on the Android side. Apple usually couples their soft- and hardware very tightly and optimizes their experience very thoroughly. As a result their hardware-wise processing requirements are lower. In this experiment however raw processing power was key and therefore it surfaced the gap clearly. Another reason for the results might be found in the OpenCV implementations themselves. Different compiler implementations were used for building the OpenCV library for both architectures.

7.2 Conclusions

As the figures of both experiments show clearly, developers and researchers need to carefully consider their implementation decisions. For example, processing keypoint detection and descriptor extraction with SIFT, SURF and BRISK requires a lot of time which users may not be willing to accept. If the application requires to process large amounts of image data it is recommended to use a client-server solution. The additional time introduced by network latency and offloading the work to a server may be the better choice, depending on the circumstances. On the other hand, should the application require processing of only one or a couple of images without the need for immediate feedback, it can be reasonable to use a local solution that runs directly on the mobile device. These recommendations are of course only valid for the single-core scenario, as it was not evaluated how the operations would perform when parallelized. Even more performance potential could lie in the inclusion of the GPU in the process. Technologies like *OpenCL*[4] or *Metal*[5] now make it easier than ever to use such resources.

Furthermore, the version of OpenCV (version 2.4.9) that was used in the experiments seems to have some memory issues in their SIFT and SURF implementations. As reported, it was not possible to execute them with images equal or above three megapixel in resolution, even not on a device with 3 GB RAM. Since a resolution of three megapixel is already quite conservative for the time this thesis is written, users either need to shrink down the images prior to the analysis (accepting a penalty of precision) or implement a client-server solution.

[4] `https://www.khronos.org/opencl`
[5] `https://developer.apple.com/metal`

CHAPTER

8 Novel Video Browsing Interfaces

In addition to the earlier already evaluated interface ideas this chapter contains interface concepts that have been published as demonstrations only[1]. Some of them use rather simple approaches while others utilize results of content analysis.

8.1 3D Filmstrip

Although the majority of movies and videos today are stored and played digitally, people when asked to describe the concept of film or video often think about an old celluloid stripe of images that was used in analog projectors and film cameras - a film strip. Therefore, the following interface concept utilizes this mental image to its advantage: a 3D Filmstrip video browser [HSB13a].

The interface projects a 3D strip into the landscape of a movie theatre. The strip is arranged in a way so that it forms rows and continues to the back (see Figure 8.1). To see more of the strip it is tilted slightly to the front in its default orientation. The strip itself is divided up into screens of equal size, similar to the individual images of a real film strip. When a video is loaded it is first uniformly cut into ten second segments and each screen on the strip a video segment is assigned. As representative keyframe of a video segment the first frame of the segment is extracted and displayed. To avoid confusion of users the

[1]Please note that the content of this chapter is adapted from [HSB13a], [HSB13b] and [HSX15b]

segments are assigned to the screens in chronological order: the first segment is assigned to the screen at the front left and the last segment is assigned to the screen at the end of the strip. Therefore, the length of the film strip is dependent on the length of the actual video.

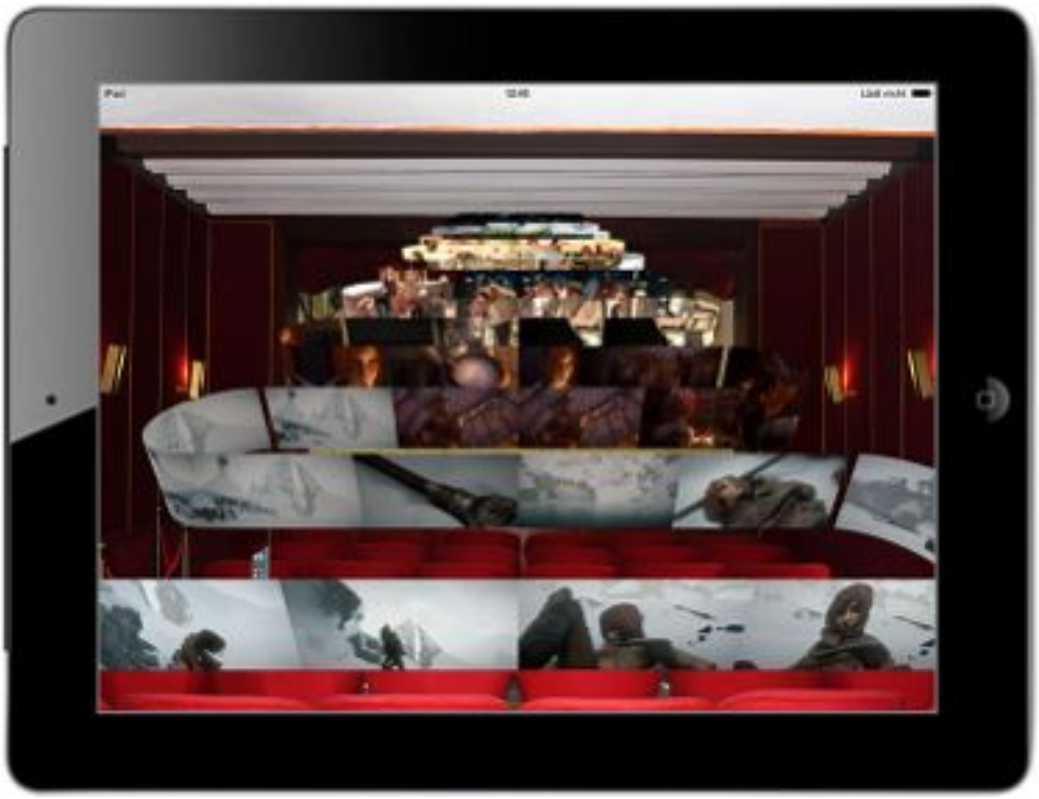

Figure 8.1: Initial view of the 3D Filmstrip.

In its default state the interface gives a good overview of the first 14 video segments, which means that approximately the first two minutes of a video are covered. To browse later parts of the video user can perform a downward drag gesture on the touchscreen. Users have to tap and hold anywhere on the screen with a single finger and drag it on the screen downwards. In this way users are able to scan through the whole strip. To scroll back to the earlier parts of the video the same motion in re is intuitive in the same way. Users just have to *push* the strip away from them. This is done by dragging the finger up (or away from users bodies).

It is also possible to change the angle at which the filmstrip is tilted. It can be done by placing two fingers at the same time on the screen and dragging them up- or downwards. See Figure 8.2 (right) for a result of an tilting operation, where the angle was increased to see more of the back of the strip.

Users are also able to manipulate the positions of the screens directly on the filmstrip. The screens can be *scrolled* sideways to the left or the right on the strip. For this, user have to place a single finger on any screen on the strip and drag it to the left or to the right. The screens then scroll accordingly. Screens that are *pushed* across the start of the filmstrip are added to the end of the film strip at the back. In accordance, screens that are pushed over the end of the filmstrip at the back are added to the start of the filmstrip at the front. Moreover, when screens change rows they are automatically adjusted so that they do not appear mirrored.

The default configuration of ten second segments might not be optimal for all kinds of videos. Therefore, users can customize the segmentation to their needs. For very long videos it might be useful to first work with rather long segments (e.g. 30 seconds, 1 minute, etc.) and then users refine their search by decreasing segment durations again. This feature can be controlled by applying pinch-gestures. To increase the segment duration (by two) users have to apply *pinch out gestures*. The strip is then refreshed with the new configuration, which can be seen in Figure 8.2 (left). To decrease segments duration by half users can use *pinch in gestures*.

Finally, users are also able to play the content of the of the visualized segments. To do this, users need to *tap* on one of the screens. The tapped screen then starts to play the video segment inside its dimensions. An additional tap on the screen pauses the playback of the segment. Fullscreen playback of video segments is also available. It can be activated by *double tapping* on a screen.

Figure 8.2: 3D Filmstrip after zoom-out operation (left) and zoom-in operation and tilting (right).

8.2 The ThumbBrowser

Smartphones and tablets are usually operated in a portrait-like orientation. Unfortunately, this way of holding the device is sub-optimal for watching videos. Videos are optimized for a landscape orientation. When displayed on screens in portrait orientation only a small part of it can be used for the actual content. This is especially problematic on the already small screens used by mobile devices.

As a consequence, smartphones and tablets are typically turned to landscape orientation. Their preinstalled video players then switch to a fullscreen playback that uses all of the available screen space. However, interacting in this orientation becomes quickly uncomfortable. Especially on a tablet the interaction with a horizontal seeker bar or playback buttons in the middle of the screen is unnecessary complicated. Users can barley reach the controls as they are too far away from their thumbs. They have to release the secure two-handed grip and use one hand for the interaction while holding the device with the other. Holding the device in such a way is tiring and unstable.

The solution is to provide users an interface that adapts to the landscape orientation. This idea is far from new. On-screen keyboards are a good example for this. The keyboard is split into two halves that are positioned on the left and the right edges of the screen so that the thumbs have no problem reaching them. Two examples can be seen in Figure 8.3.

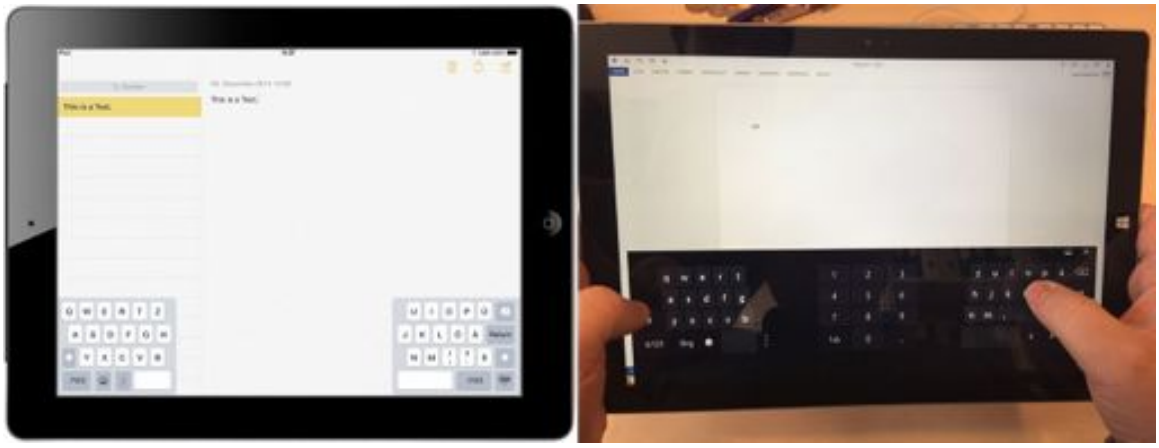

Figure 8.3: Split keyboards on iOS (left) and Windows 8 (right).

Nevertheless, video players have not picked up this idea yet, although watching video is one of the best reasons to hold devices in landscape orientation. Therefore, in the following a video player is proposed that focuses on usage in landscape mode: the ThumbBrowser. Users are able to operate it with their thumbs only, without the need to adjust hand position. Furthermore, it incorporates hierarchical browsing features, bookmarking and, in a current unpublished extension, on-the-fly background analysis of face occurrences and color signatures of all shots of a video. An interim milestone was presented at the ACM International Conference on Multimedia 2013 and shortlisted for the best demo award [HSB13b].

8.2.1 Interface

In normal playback mode all available screen space is utilized for the video and no controls are shown so that users can concentrate on the content. To use the interface controls users have to place their right or left thumb on the screen. This activates, dependent on the thumb used, different interface controls. The right thumb activates a *vertical seeker control* to quickly scan through the video and jump to any position. The left thumb activates a *radial menu* that offers functionality for play/pause, fast forward and fast rewinding. For an image of the interface please see Figure 8.4.

The *vertical seeker control* consists of a vertical timeline and a magnifying glass with additional information areas, as can be seen in Figure 8.4 on the right side. The top of

Figure 8.4: The default view of the initial version of the ThumbBrowser with its two main controls: a radial menu for the left thumb and a vertical seeker control for the right thumb.

the timeline is mapped to the beginning of the video, the bottom to the end of the video. A small marker on the timeline tells users their current video position. The magnifying glass follows the position of the thumb as the user drags it up or down the timeline. When users make horizontal movements the magnifying glass follows the thumb so that its blue handle always stays directly below it. The magnifying glass shows users the corresponding frame that resides at its current location on the timeline. This enables users to quickly scan through the content of the video. In the top bar of the control, a label shows the time code of the current position of the magnifying glass.

Should users decide that they want to navigate the video to the current position of the magnifying glass, they can lift their right thumb. This triggers the video player to seek directly to the selected position. In parallel, the whole control is faded out. In the case that

users do not want to seek to a certain position, they can drag the magnifying glass all way to the right and lift their thumb. This cancels the seek process and hides the control. The abortion is also visually indicated by a red colored area in the top bar (see Figure 8.5).

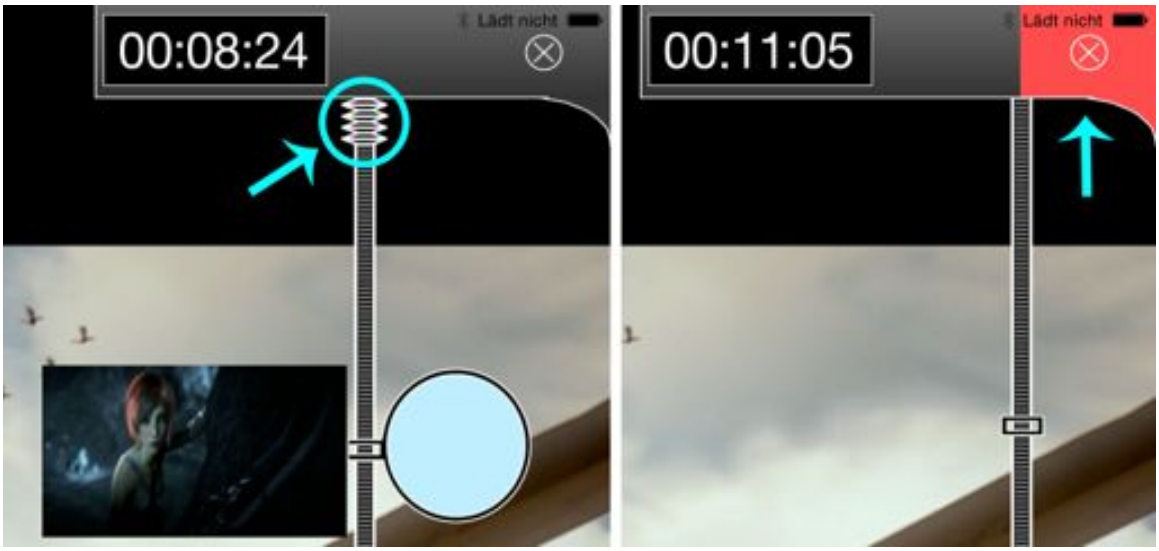

Figure 8.5: Squeezing-effect of the timeline (left) and visual indication for aborting a jump (right).

The *radial menu* on the left side of the screen offers in total five options: play/pause, fast forwarding, fast rewinding, timeline zoom, video bookmark. To activate an option, users have to place their left thumb on the screen and drag their finger over the respective option. Furthermore, during fast forwarding and fast rewinding the vertical seeker control is also displayed for orientation reasons.

The *timeline zoom* option (indicated by a magnifying glass in the menu - see Figure 8.4) is intended to give users finer control over the vertical seeker control. The problem arises when users want to navigate in long videos. Since the complete length of a video has to be mapped to the area of the timeline, the longer the video is, the harder it gets to navigate on a fine granularity level (e.g., second by second). Users might have the feeling, that the content they are looking for should be around four minutes into the video but they now have problems further examining the video at that time range. For such a case the interface

offers *timeline zooming*. When activated it remaps the timeline to an area of +/- 20 seconds around the current position of the magnifying glass. This makes navigation on a detailed level much easier. To make it clear to users that they have activated this mode the top and bottom part of the timeline gets *squeezed* in an animation (as can be seen in Figure 8.5). When the video continues to play the mentioned zoom-window of the timeline is adjusted accordingly, so that the current position is always located in the center of it. The zoom process can also be repeated to get even finer navigation control. Users can also return to an earlier zoom level by choosing an available zoom out option.

The ThumbBrowser records every seek process in the video to give users a way to return to the last position. This functionality is helpful when users just want to check what happened earlier in the video (for reference) and then try to return to their original playing position to continue watching. To activate this feature, users need to swipe into the screen from its left edge. Moreover, users can manually add bookmarks by using the apropriate option of the radial menu (indicated by a star symbol - see Figure 8.4).

8.2.2 ThumbBrowser V2

Based on the feedback received at the ACM Conference on Multimedia 2013 with the original version, a set of improvements was implemented. The first improvement focused on the extended vertical seeker control. While it was received well for its hierarchical approach to scan through portions of the video in great detail very quickly, some had trouble with the physical interaction pattern. Therefore, the visualization of the zoom process is redesigned.

In contrast to the original version the view of the extended seeker control is now changed to a scrollable vertical strip of keyframes (see Figure 8.6). Two markers on the timeline represent the current position in the video (black) and the position of the keyframe strip (blue). Initiating a seeking process can be performed by tapping on one of the frames. When users start the playback of the video the strip is scrolled accordingly in a way so that the videos current playback position is always in the middle of the filmstrip.

Figure 8.6: The new controls of the improved ThumbBrowser: Extended timeline as well as strip-like visualization (left) and the new bookmark pane (right).

Moreover, in the new version of the ThumbBrowser a bookmark control element was added. It can be displayed by swiping with the left thumb into the screen (see Figure 8.6). It provides an additional button for creating a new bookmark at the current position. Furthermore, a scrollable list of all available bookmarks is displayed. Each entry of the list shows the frame of the bookmark, the timecode and the date when the bookmark was created. To navigate to one of the bookmarks users just have to tap it. It is also possible to delete a bookmark by swiping a finger across an entry to the left.

The last improvement concerns adding simple means of content analysis to further enhance the search experience of users. Face detection is added to the interface for search scenarios where users want to find scenes with or without people. For example, if users want to find an interview in a documentary about nature or animals this feature offers visual aid. In such videos people only appear in a few places. On the other hand, it could also help to find landscape scenes in videos where a lot of people appear, like in news videos. The detection happens in the background and starts when a video is opened the first time. The actual user interaction is never interrupted for this. Found places are added on-the-fly as they are detected. The found positions are also cached so that the process does not have to start fresh every time the video is re-opened. Found places are visualized in the timeline of the extended seeker control. Small blue markings on the left side indicate that faces where

found at a position. The more pronounced the marking is, the more faces were found.

Moreover, frames of the video are sampled in a uniformly way and their dominant color is extracted. The detected color is then displayed as a small marking on the right side of the timeline. This helps to find scenes in the video with a distinct color setting, like announcements, outdoor scenes or similar. To see an example of this visualization please refer to Figure 8.7.

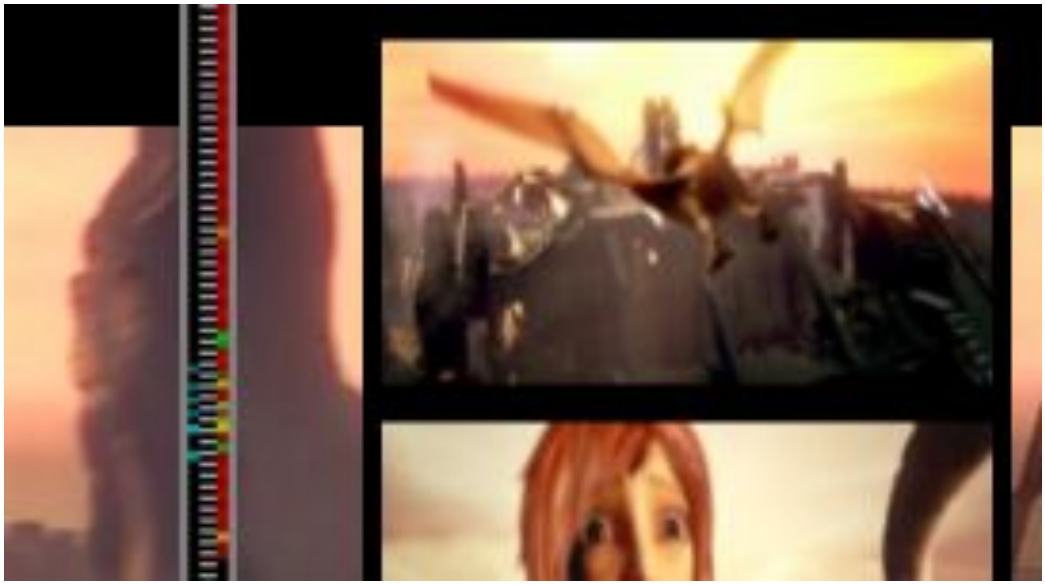

Figure 8.7: Markings on the new timeline indicating occurrences of faces (left side of the timeline) and dominant color (right side of the timeline).

8.2.3 Further Development

Partly inspired by the original version of the *ThumbBrowser*, in a recently published study by Hürst and Hoet [HH15] a storyboard-based video browser is compared to a thumb-optimized version. Unfortunately, despite positive comments and ratings by users the thumb-optimized browser could not showcase a significant improvement over the storyboard design. The authors also note, that only few of their participants actually took advantage of the thumb-orientated controls and instead used them traditionally, with one hand holding the device, interacting with the other one. As often the case, it might just be that users were already

too accustomed to the traditional way video browsers are operated. It would be interesting to have study participants take part of an extensive training and then only allow them to operate the interface in the intended way. In future work, these findings will be considered and further explored. Furthermore, it might also be advantageous to adapt the interface for smartphone screens, as usage patterns could be totally different.

8.3 The Multi-Stripe Video Browser

Based on the work of the earlier described KNT-Browser a new mobile video browser for taking part in the 2015 *Video Browser Showdown*[2] (VBS) was designed: The Multi-Stripe Video Browser (MSV-Browser) [HSX15b]. The VBS is an annual live video search competition. It gives researchers an opportunity to evaluate and demonstrate their video search tools.

The browsing system consists out of two parts: a server part and a mobile part. The server part performs content analysis for the following features: detection of shot and sub-shot boundaries, color analysis, motion analysis, keypoint analysis, face detection and selection of so called *templates*, which will be described in detail later.

The mobile part consists of an app written for the Apple iPad. It features a browsing area and a filtering area. Most prominent element of the interface are the two navigation bars at the lower half of the screen. The bars follow the same principle that was used in the KNT-Browser. Each displayed keyframe represents a detected sub-shot. In this implementation however, the middle navigation bar was removed. In the evaluation of the KNT-Browser it became clear that the middle bar was rarely used by study participants. Therefore, it made sense to reduce the configuration to an overview navigation bar with slim slots but good overview and a detail navigation bar with full-size keyframes.

[2] `http://www.videobrowsershowdown.org`

8.3.1 Analysis Component

As mentioned the analysis of videos is performed in an offline manner. No network connection between the server and the mobile application is necessary. In a first step, shots and sub-shots are detected by looking at changes in terms of motion and in terms of color changes. For this, the method shown by Luo et al. [LXSS14] was used. After that, the frames of the individual shots are further investigated. Each frame is partitioned into nine spatial areas of equal size. For each of the areas a color histogram is generated and the keypoint density is detected. The color histogram that was used is based on the HSV color space and supports 19 bins (16 colors as well as white, gray and black). Furthermore, the density of keypoints is computed based on Shi-Tomasi corner detection [ST94]. This is performed for all frames of a shot. The color and keypoint features are averaged for the corresponding area for all frames of the shot, which generates a summarized description of a shot in terms of color and keypoints.

The shots are moreover analyzed regarding motion. For each shot a motion histogram is generated. It consists out of eight bins using four different motion directions (90 degree each) as well as the corresponding average motion speed. Furthermore, the presence and the count of faces in shots are detected. For this, the face filter that is part of Apples Core Image framework was utilized. It was used in its default configuration. The videos are also analyzed for occurrence of prominent concepts or *templates*. The detected templates are used to provide unexperienced users a faster way to browse and filter multiple videos.

Finally, saliency detection is applied on the representative shot keyframes in the navigation bars. For this, the method shown by Achanta et al. [AHES09] is applied. It ensures that the most important part of a keyframe will be visible when visualized in the overview navigation bar, as only a narrow stripe will be visible. All of the gathered shot information of a video is recorded into a metadata-file. It is uploaded to the tablet together with the (size-reduced) video files.

8.3.2 Mobile Component

The mobile component is implemented as an iPad app that loads the appropriate videos and the corresponding metadata files generated by the server component. The interface can be divided into three areas as can be seen in Figure 8.8. Area A is focused on previewing content as well as providing typical video player controls. It also contains buttons for configuration of the VBS server data and submitting the video and timecode that is currently displayed in the preview window.

Figure 8.8: Main interface of client app with interface areas highlighted.

Area B focuses on filtering-controls. As soon as the user sets any option the app filters all video sequences for matches and displays the results. The filtering process is visualized via a short period of dimming and locking the interface, as well as a flashing message in the status bar at the bottom (see Figure 8.8).

The region all way to the left of area B in Figure 8.8 contains controls used for setting a custom color layout for the nine spatial areas earlier described (color layout grid). To set

a color for one or multiple areas users have to tap on the areas and then choose color from the palette on the right. Furthermore, it is possible to reset a setting with the appropriate option in the palette. In a similar fashion the keypoint density can be set for each of the nine areas, by choosing a density option below the color palette: high (H), medium (M) or low (L).

Moreover, three buttons below the color layout grid can be used to set different values for color and keypoint density of up to three successive video sequences. Only pairs or triples of video sequences that match the filtering criteria in that order will then be displayed.

Below the color layout grid the motion filtering control is available. It consists out of a quadrisected circle for setting options of upwards, downwards, left and right motion, and a bar that can be set to set the amount of no (still) motion. User can set a motion filter by simple drag gestures.

To the right of the motion filter controls two sliders and two buttons allow users to set the filtering accuracy/threshold (T), face count (F), apply a preset to filter for zoom motions (button with four arrows pointing to the corners) and to access the template gallery (button with icons).

The template gallery button enables users to switch to a view where all available templates can be browsed (see Figure 8.9). Each thumbnail in the gallery represents a detected template. Furthermore, a counter below the thumbnail informs users about how often the template was found in the video collection. A single tap on one of the thumbnails returns users to the detail view of the interface, preloaded with all video segments that match the chosen template.

Area B also offers users a button that activates filtering via a captured photo (at the left middle side of Figure 8.9). Pressing the button switches the view to a view finder for taking the photo. After that, it is possible to crop the taken image. The cropped image is then analyzed regarding its color layout. When complete the view switches back to the detail view, preloaded with matching video shots.

Area C focuses on browsing the found video shots by utilizing two navigation bars based on the work done in the KNT-Browser. Furthermore, users have the option to initiate similarity filtering. This can be activated by long pressing on one of the keyframes. A then

Figure 8.9: Template gallery of the MSV-Browser.

appearing popup-menu provides a button to activate filtering for similar video shots in comparison to the selected one. The video shots are compared in terms of their color layout values, keypoint density and movement data.

Finally, it is also possible to switch the view into an *overview mode*. It can be activated by simply turning the device to portrait orientation (see Figure 8.10). In this mode each video gets its own navigation bar. The list of navigation bars can be scrolled horizontally via drag gestures. This mode can be especially useful after a filtering operation. Since more keyframes are visible the results can be processed by users even faster. When a keyframe of any navigation bar is tapped the interface returns to the default detail view with the touched keyframe visible in the preview player as well as appropriately positioned navigation bars. It is possible to return to detail view by simply turning the device back to landscape orientation.

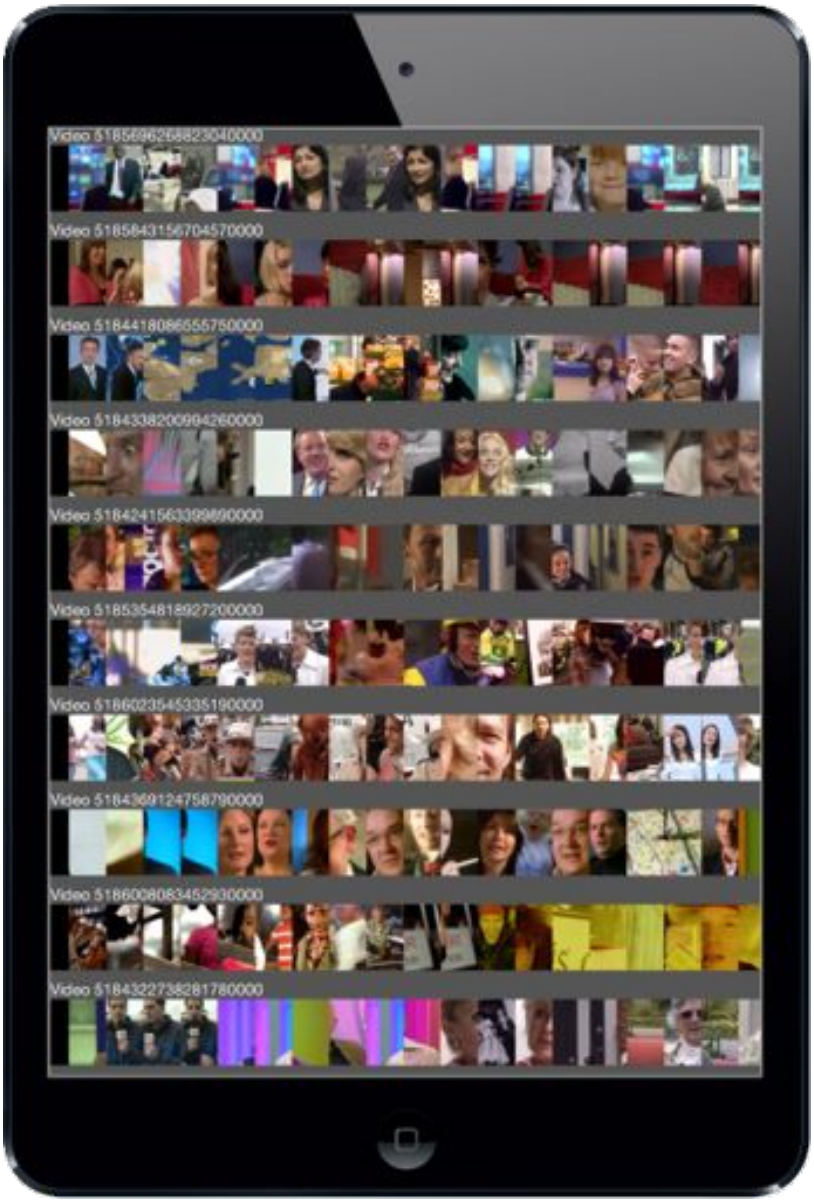

Figure 8.10: MSV-Browser's overview mode activated by turning the device to portrait orientation.

CHAPTER

9 Conclusions

Smartphones and tablets will accompany our life for a while. The same is true for the desire of people to capture and enjoy images and videos. Steve Jobs once said in an interview at the D8 conference, that he expects a future of computing similar to todays ratio of trucks and cars:

"When we were an agrarian nation, all cars were trucks, because that's what you needed on the farm. But as vehicles started to be used in the urban centers, cars got more popular. Innovations like automatic transmission and power steering and things that you didn't care about in a truck as much started to become paramount in cars. ... PCs are going to be like trucks. They're still going to be around, they're still going to have a lot of value, but they're going to be used by one out of X people." [D10]

With such expectations also the need for improving the image and video experience on those smart devices will probably not diminish. For that, we need to understand the needs and expectations better and continue to perform research in this area. The work in this thesis tried to give a couple of indications of how better image and video browsing could look like.

In Chapter 3 it was discovered that users approximately store between 100 and 400 images on their devices in average. Furthermore, they store more on their smartphone than on their tablets. This is interesting, as they seem to buy tablets with higher storage capabilities

in comparison with smartphones. Also, they generally do not organize or backup their mobile image and video collections in any way. Interestingly, women seem to store significantly more images on their smartphones than men. Also, it looks like the software environment has an influence on the image count, as iOS users store significantly more images on their smartphones than Android users. Some expectations of us were confuted, as we could not see any relation between storage capabilities and image count or between capture frequency and image count. Furthermore, users store only a very limited number of videos locally on their smartphones and tablets. Being ware of these insights, new image and video browsing concepts are shown in the following chapters that respect these figures.

Chapters 4 and 5 report about two large user studies investigating new image browsing interfaces for tablets and smartphones. The first concentrates on finding out if there are indications whether the used shape of 3D color sorted interfaces matters in terms of users' search performance. Interestingly, in the test users were as fast with a 3D globe as they were using a 3D ring. The second study focuses on transitioning the 3D interfaces from tablets to smartphones as well as comparing them with two additional 2D color sorted interfaces: a typical image grid and an image pane. Contrary to earlier studies on tablets, the 3D interfaces could not deliver significant improvements in search time. This could imply that the utility of such interfaces is related to the available screen space. Furthermore, the rather simple idea of the image pane could provide equal access times to images, regardless of their sorting position.

Mobile video browsing is the focus of Chapter 6 by presenting the KNT-Browser. It focuses on improving the browsing and search experience in a single video. For this, the notion of sub-shots is introduced. Interface-wise it provides easy navigation through three stripes of keyframes that represented detected sub-shots in the video at different levels of detail. In the evaluation the KNT-Browser successfully outperformed the default video player at KIS-like tasks significantly in terms of search time. It also produced about 50% less erroneous trials than the default video player. Moreover, the KNT-Browser also succeeded in the questionnaires as it performed significantly better in all categories of the NASA TLX workload index and it was the preferred interface for most of the study participants.

Evaluating content analysis performance on smartphones and tablets is described in Chapter 7. At that time state-of-the-art iOS and Android devices were tested in two experiment setups. Popular analysis functions of the OpenCV library were benchmarked on how well they performed in terms of execution time and battery requirements. It has been shown that mobile devices can reach up to 80% of the performance shown by a consumer grade laptop. Nevertheless, the utilized methods have to be chosen wisely. In very demanding cases (e.g., batch detection of SIFT keypoints) it is recommended to consider a client-server approach, as network latency could undercut local execution time.

Finally, a short digression into published but not yet evaluated work was discussed in Chapter 8. With the *3D Filmstrip* an engaging new way how to visualize video content was presented by using the metaphor of a traditional celluloid film strip. The *ThumbBrowser* was introduced with a new more natural and more comfortable way of watching and browsing videos in landscape orientations. It focuses on being used with the left and right thumbs. Furthermore, in a later iteration, it features background and on-the-fly analysis of faces and dominant color. In the end the *Multi-Strip Video Browser* was presented. It features various ways how to filter and browse whole video collections via color layout, keypoint density, motion and shot similarity.

9.1 Future Research

The ideas for future research that came up during the work on this thesis are extensive. Some of the most promising ones will be discussed here.

- Utilizing 3D visualizations in the browsing domain still has huge potential. Only few works focus on using it for video browsing. It could be used to better visualize the different aspects of video by exploiting the third dimension.
- As it was already discussed in chapter 8, currently an improved version of the ThumbBrowser is in work that will incorporate keyframe analysis features. It would also be

interesting to evaluate it not only with entertainment content but also with videos of special domains, like security footage or medical recordings.

- During the lifespan of the NGVB project other researchers of the field proposed using concept recognition for image browsing on smartphones. As most of the projects prototypes use color sorting it would be interesting to make a comparison between the color sorting, concept sorting and a combination of both.

- Furthermore, new device types are coming up that could also be important for the field of image and video browsing. Some manufacturers have started to equip their smartphones with mini projectors and researchers are already exploring how this technology could be used in new ways [Kau14]. It is easy to imagine that this could also be used for new ways of image and video browsing.

- Currently, multiple manufacturers extensively push smart watches. They are intended to provide unobtrusive notifications and most of them are currently tightly coupled to a smartphone. Nevertheless, in most of their demonstrations images also play a role (e.g., favored images are automatically synced between Apple Watch and iPhone). How this will further develop in the future remains to be seen.

- Another push is currently visible for augmented and virtual reality glasses. Microsoft announced recently their own headset for augmented reality (HoloLens[1]) and showcased purposes for entertainment as well as work. Oculus Rift[2] and Sony's Morpheus[3] virtual reality glasses enable a level of immersion that was not possible until now. Such hardware could be used to enable completely new ways of interacting with images and videos.

[1] `http://www.microsoft.com/microsoft-hololens/en-us`
[2] `https://www.oculus.com`
[3] `http://www.bbc.com/news/technology-31723030`

9.2 Closing Words

Niépce and Muybridge (Figure 9.1) would probably never have thought, that there will be a time, when people will own so many images and so many and long videos, that it causes headaches to actually find something again. For pioneers it is often difficult to envision what implications their inventions will have for the future.

Figure 9.1: Joseph Nicéphore Niépce (left) and Eadweard Muybridge (right).

Today, we live in such a world. Never before is so much content produced and consumed, for entertainment as well as for work. Technology will have to advance in order to be able to cope with such loads of content in an effective and efficient way.

APPENDIX

A OpenCV Performance Analysis on iOS

In this first step of evaluating the OpenCV performance the focus is on iOS devices[1]. Function calls that users typically utilize in their projects were chosen and set up. Furthermore, a representative dataset was prepared. First the experimental setup will be reported. After that the results of the measurements as well as a first short discussion follow.

A.1 Setup

The tested OpenCV operations are grouped into three measurement phases. In the first phase typical operations used in the computer vision area are tested, like blurring an image using Gaussian blur, detecting faces and detecting edges. In the next phase functions calls to common keypoint detection and descriptor extraction operations are grouped, like SIFT [Low04] and SURF [BETG08]. The last phase consists out of operations that match descriptors extracted from two consecutive frames of a video.

For the measurements of the first two phases, 5000 images[2] were randomly drawn from the MIRFLICKR25000 dataset [HL08]. The images differ in resolution and have an average of 463 horizontal and 397 vertical pixels. Each OpenCV-function call is tested with each of these images five times in a row. The measured times are then averaged. This is done in

[1] Please note that the content of this chapter is adapted from [HCS14]

[2] At `http://www.hudelist.org/research/phdthesis/5000MIRFLICKRListing.txt` you can download a filename-listing of all used images.

order to even out measurement differences caused by unpredictable interventions of the OS. The time measurements for each of the images are then summed up and averaged to get an overall performance measure for each applied OpenCV-function.

The last phase, which concentrates on descriptor matching, uses a different dataset than the phases before. The first 5000 frames of the *test video 031* of the Video Browser Showdown 2013[3] [SAB+14] are extracted (a recording of a Dutch news journal). The video is encoded in H.264 with an average bitrate of 619.7 kBit/s and a resolution of 640 x 360 pixels. The tested matching approaches differ in terms of the used descriptors, but all use the same brute force matcher that is part of the OpenCV framework, called *BFMatcher*. In this phase, only the actual matching process is measured, not the keypoint detection and descriptor extraction. Each matching approach has to find matches between two consecutive frames, starting with the first and second frame in the video, continuing with the second and third frame, until the end of the video. Similar to the earlier phases, each matching process of a frame pair is repeated five times and the measured times are averaged. The mean of the averaged times of all frame matches are then computed to get the overall measurement for each matching approach.

For the experiment, different generations of Apple's iPads and iPhones were used, including their (at that time) latest and fastest units, the iPad Air and the iPhone 5S. To compare the performance to a standard PC, the performance of a MacBook Pro 13" with Retina Display (late 2012 version) with an Intel Core i5 at 2.5 GHz, 8 GB of RAM and integrated Intel HD Graphics 4000 was measured additionally. For a listing of the specifications of the devices please see Table A.1. On all of the mobile devices the latest available OS version at that time was used: iOS 7.0.4. The MacBook Pro used Mac OS X 10.8 Mountain Lion. Moreover, version 2.4.7 of the OpenCV framework was installed and used on all of the devices.

[3] `http://www.videobrowsershowdown.org`

Device	CPU Clock	CPU Cores	CPU Architecture
MBP 13" (2012)	2.5 GHz	2	Intel Core i5 3210M (64 Bit)
iPad Air	1.4 GHz	2	Apple A7 (64 Bit)
iPad 4	1.4 GHz	2	Apple A6X (32 Bit)
iPad 3	1 GHz	2	Apple A5X (32 Bit)
iPad Mini 1	1 GHz	2	2. Gen. Apple A5 (32 Bit)
iPad 2	1 GHz	2	Apple A5 (32 Bit)
iPhone 5S	1.3 GHz	2	Apple A7+M7 (64 Bit)
iPhone 5	1.3 GHz	2	Apple A6 (32 Bit)
iPhone 4s	800 MHz	2	Apple A5 (32 Bit)

Table A.1: Specification breakdown

A.2 Results

First, common OpenCV operations like grayscaling and blurring are reported. After that, the sub-chapter continues by reporting the performance of keypoint detection and descriptor extraction operations. Finally, the results of the descriptor matching phase are reported.

A.2.1 Common OpenCV Operations

In this part it is evaluated how long it takes to:

- Grayscale an image
- Blur an image with Gaussian Blur
- Detect faces in an image
- Calculate the RGB and HSV histograms
- Detect edges using Canny edge detection [Can86]

For Gaussian Blur the *GaussianBlur(...)* function is used with a *kernel size* of 21×21 and a *sigma* of *8.0* to produce recognizable blurred result images (average image resolution: 463×397 pixel). The face detection is realized by using a trained *CascadeClassifier* and its *detectMultiScale(...)* function. Images are converted into grayscale before the actual

detection takes place. A *scaleFactor* of *1.1*, *minNeighbors* of *2* and a *minimum size* of *30* × *30* are used. The RGB histograms are calculated with a size of 256 bins, a range from 0 to 256 and the *uniform-flag* set to *true* and *accumulate* set to *false*. The HSV histograms are calculated with 30 hue levels and 32 saturation levels with the standard ranges. The default values for *uniformity* (*true*) and *accumulation* (*false*) are used. For the edge detection using *Canny* the image is first grayscaled and then blurred with a *kernel size* of *5* × *5* and a *sigma* of *1.2*. The *Canny(...)*-function is then measured with thresholds one and two set to 0 and 50 respectively. For a breakdown of the measurement results please see Table A.2.

Device	Grayscale	Gaussian Blur	Face Detection	RGB Hist.	HSV Hist.	Canny Edge Detection
MBP 13" (2012)	0.26	3.97	134.27	0.40	0.30	3.10
iPad Air	0.32	42.31	214.10	2.48	0.89	9.34
iPad 4	0.58	80.66	282.30	2.47	1.26	15.49
iPad 3	1.18	153.64	502.37	2.88	3.10	26.22
iPad Mini 1	1.17	151.19	505.66	2.94	3.17	26.14
iPad 2	1.15	153.30	500.07	2.91	3.14	26.11
iPhone 4s	1.30	190.00	620.62	3.55	3.84	32.10
iPhone 5	0.61	86.32	296.73	2.63	1.35	16.39
iPhone 5S	0.34	45.28	235.56	2.49	0.92	9.73

Table A.2: Average execution times in ms of operations applied to an image (common group).

A.2.2 Keypoint Detection/Descriptor Extraction

For keypoint detection and descriptor extraction, several popular algorithms that are built-in in the OpenCV framework are evaluated. The measurement results of Table A.3 include the time needed to detect keypoints as well as to extract descriptors based on the detected keypoints. The algorithms that are evaluated in this phase are ORB [RRKB11], BRIEF [CLSF10], BRISK [LCS11], SIFT, SURF, FREAK [AOV12], and FAST [RD06]. In case of BRIEF and FREAK the descriptor extraction process is combined with GoodFeaturesToTrack [ST94] to detect keypoints.

Device	ORB	BRISK	FREAK	FAST	BRIEF	SIFT	SURF
MBP 13" (2012)	10.73	248.14	17.04	1.07	1.87	172.71	341.32
iPad Air	52.37	775.70	46.49	7.72	3.64	877.09	805.50
iPad 4	80.30	1167.24	84.48	15.03	6.89	1378.78	842.3.0
iPad 3	162.99	2155.05	184.84	24.74	17.32	3518.18	1606.63
iPad Mini 1	162.96	2156.24	185.66	24.86	17.63	3586.21	1627.40
iPad 2	162.23	2150.66	184.24	24.70	17.20	3515.13	1599.75
iPhone 4s	200.54	2673.05	222.74	30.59	20.77	4320.10	1980.38
iPhone 5	84.41	1232.42	89.92	16.11	7.42	1474.19	804.29
iPhone 5S	55.15	829.54	49.04	8.22	3.76	1028.17	1024.97

Table A.3: Average execution times in ms of operations applied to an image (keypoint detection/extraction group).

Device	SIFT	SURF	BRISK	BRIEF
MBP 13" (2012)	51.98	33.98	2.11	15.69
iPad Air	254.24	163.93	3.23	23.63
iPad 4	394.12	204.11	6.68	37.78
iPad 3	1671.09	746.41	14.38	76.103
iPad Mini 1	1643.27	756.31	14.47	76.08
iPad 2	1610.51	744.96	14.72	75.89
iPhone 4s	1888.81	918.46	17.94	94.405
iPhone 5	421.38	243.62	8.43	40.46
iPhone 5S	390.00	223.53	3.74	32.52

Table A.4: Average execution times in ms of operations applied to an image (keypoint matching group).

A.2.3 Descriptor Matching

In the matching phase the performance of SIFT, SURF, BRISK and BRIEF descriptors are evaluated regarding how fast they can be matched for two successive frames of a video. For each of the frames the preceding process of keypoint detection and descriptor extraction is performed but excluded from the actual measurement. For the actual matching *BFMatcher* is used, a brute force matcher. For a breakdown of all results please see Table A.4.

A.3 Discussion

The comparison of smartphones and tablets with a regular consumer grade laptop revealed some interesting results. From the start it was clear that the laptop would outperform the mobile devices. Nevertheless, it can be seen that these devices are able to deliver up to 80% of the laptops' performance in selected scenarios. Functions like grayscaling, face detection or canny edge detection perform rather well on those devices, given the constraints under which they have to operate (see Figure A.1). Interestingly, histogram creation and Gaussian Blur are not so fast on mobile devices in comparison to a traditional computer. Gaussian Blur performed especially bad, with requiring ten times as long as the laptop on the fastest mobile device in the setup. Furthermore, a rather strange behavior is visible regarding RGB histogram creation. Typically, it requires significant more time than HSV histogram creation. However, on some older devices this constellation is reversed. Speculating about the reasons for this is rather hard. It might be, that newer devices incorporate some hardware optimization that speeds up the calculations required for HSV histograms.

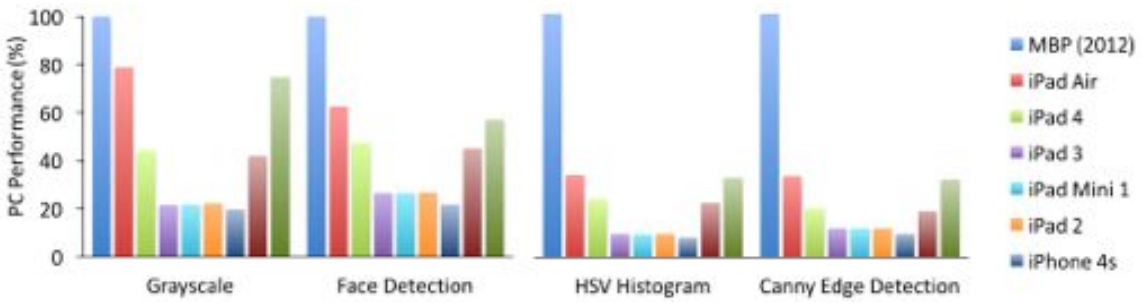

Figure A.1: Results of common operations.

In terms of keypoint detection and descriptor extraction, the top devices can provide up to 50% the performance of the laptop in case of BRIEF. Moreover, they can provide up to 40% in case of SURF (see Figure A.2). Descriptor matching showed a rather good performance on the iPad Air and on the iPhone 5S with BRISK and BRIEF descriptors. The devices are able to provide about 65% of the performance the laptop computer (see Figure A.3). In case of matching SIFT and SURF descriptors, the results are not that good with only 20% of performance of the consumer grade laptop.

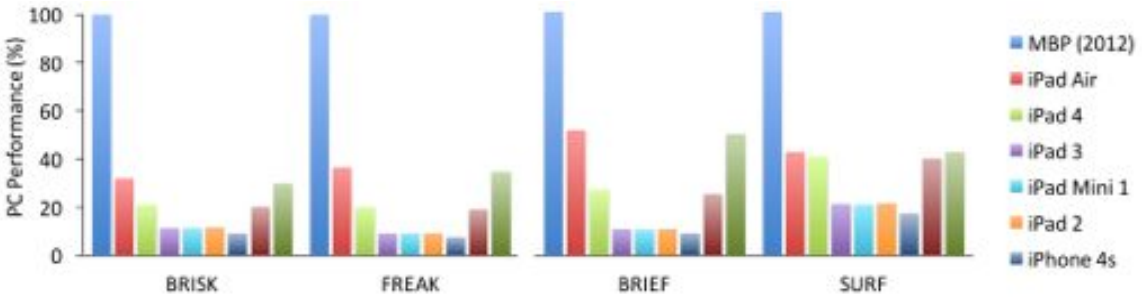

Figure A.2: Top results of keypoint detection and descriptor extraction.

Furthermore, the very similar performance of iPad 3, iPad 2 and iPad Mini should be noted. This can be seen in all the measurements that were performed. When examining the CPUs of the devices it becomes clear that they all use very similar incarnations of the same chip: a dual-core A9 processor with 1 GHz. The iPad 3 uses a slightly improved version (A9X) but the improvements mainly concentrated on the integrated GPU (one vs. four cores) that had to support the higher resolution Retina Display. In fact, this might even had a slight negative effect on raw CPU performance, e.g. increased overhead. In the measurements the iPad 3 is sometimes slightly outmatched by its predecessor. The the increased GPU performance was not utilized, as only log data was displayed on the screens during execution time.

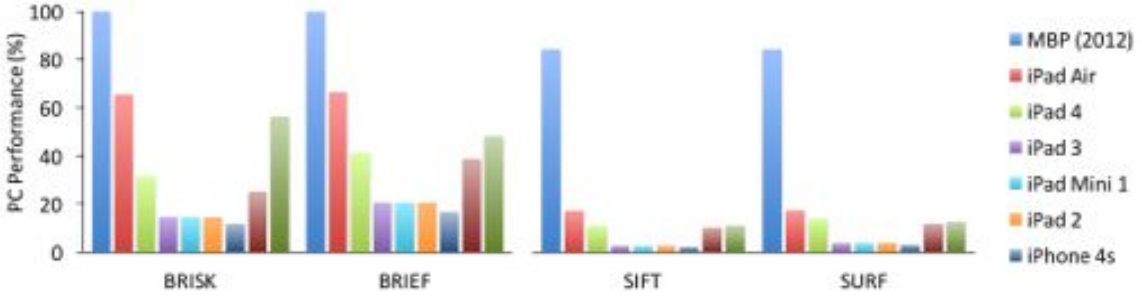

Figure A.3: Results of the matching measurements.

In general it can be said that the tested devices are in fact absolutely potent enough to run certain content analysis tasks, although users should expect increased waiting times. Especially SIFT, SURF and BRISK should be applied with the user in mind. Moreover,

face detection can have a notable impact. Asynchronous and background execution are a must to avoid an unpleasant experience.

Bibliography

[AHES09] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk. Frequency-tuned salient region detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1597–1604, June 2009.

[AHSS12] David Ahlström, Marco A. Hudelist, Klaus Schoeffmann, and Gerald Schaefer. A user study on image browsing on touchscreens. In *Proceedings of the 20th ACM international conference on Multimedia*, MM '12, pages 925–928, New York, USA, 2012. ACM.

[AOV12] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517, 2012.

[BCD+12] R. Borgo, M. Chen, B. Daubney, E. Grundy, G. Heidemann, B. Höferlin, M. Höferlin, H. Leitte, D. Weiskopf, and X. Xie. State of the art report on video-based graphics and video visualization. *Comp. Graph. Forum*, 31(8):2450–2477, December 2012.

[Bed01] Benjamin B. Bederson. Photomesa: A zoomable image browser using quantum treemaps and bubblemaps. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, pages 71–80, New York, NY, USA, 2001. ACM.

[BETG08] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346 – 359, 2008. Similarity Matching in Computer Vision and Multimedia.

[BHH+10] Shelley Buchinger, Ewald Hotop, Helmut Hlavacs, Francesca De Simone, and Touradj Ebrahimi. Gesture and touch controlled video player interface for mobile devices. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 699–702, New York, NY, USA, 2010. ACM.

[BZP10] Andrei Bursuc, Titus Zaharia, and Françoise Prêteux. Mobile video browsing and retrieval with the ovidius platform. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 1659–1662, New York, NY, USA, 2010. ACM.

[BZP12] Andrei Bursuc, Titus Zaharia, and Françoise Prêteux. Ovidius: A web platform for video browsing and search. In Klaus Schoeffmann, Bernard Merialdo, AlexanderG. Hauptmann, Chong-Wah Ngo, Yiannis Andreopoulos, and Christian Breiteneder, editors, *Advances in Multimedia Modeling*, volume 7131 of *Lecture Notes in Computer Science*, pages 649–651. Springer Berlin Heidelberg, 2012.

[Can86] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, Nov 1986.

[CBH+12] C. Czepa, S. Buchinger, H. Hlavacs, E. Hotop, and Y. Pitrey. Towards an energy-efficient attention-aware mobile video player with sensor and face detection support. In *2012 IEEE International Symposium on aWorld of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6, June 2012.

[CHDF14] Claudiu Cobarzan, Marco A. Hudelist, and Manfred Del Fabro. Content-based video browsing with collaborating mobile clients. In Cathal Gurrin, Frank Hopfgartner, Wolfgang Hurst, Hovard Johansen, Hyowon Lee, and Noel O'Connor, editors, *MultiMedia Modeling*, volume 8326 of *Lecture Notes in Computer Science*, pages 402–406. Springer International Publishing, 2014.

[CHSP15] Claudiu CobÃ¢rzan, Marco A. Hudelist, Klaus Schoeffmann, and ManfredJÃ¼rgen Primus. Mobile image analysis: Android vs. ios. In Xiangjian

He, Suhuai Luo, Dacheng Tao, Changsheng Xu, Jie Yang, and Muhammad-Abul Hasan, editors, *MultiMedia Modeling*, volume 8936 of *Lecture Notes in Computer Science*, pages 99–110. Springer International Publishing, 2015.

[CLSF10] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision - ECCV 2010*, volume 6314 of *Lecture Notes in Computer Science*, pages 778–792. Springer Berlin Heidelberg, 2010.

[Coh73] J Cohen. Eta-squared and partial eta-squared in communication science. *Human Communication Research*, 28(473-490):56, 1973.

[D10] John Paczkowski All Things D. Apple CEO Steve Jobs Live at D8: All We Want to Do is Make Better Products. `http://allthingsd.com/20100601/steve-jobs-session/`, 2010. [Online; accessed 28-May-2015].

[dRSW08] Ork de Rooij, Cees G.M. Snoek, and Marcel Worring. Balancing thread based navigation for targeted video search. In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, CIVR '08, pages 485–494, New York, NY, USA, 2008. ACM.

[Dun64] Olive Jean Dunn. Multiple comparisons using rank sums. *Technometrics*, 6(3):241–252, 1964.

[Fli14] Flickr. Camera Finder - Most Popular Cameras in the Flickr Community. `https://www.flickr.com/cameras`, 2014. [Online; accessed 11-Sep-2014].

[Gan12] Roman Ganhör. Propane: Fast and precise video browsing on mobile phones. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, MUM '12, pages 20:1–20:8, New York, NY, USA, 2012. ACM.

[Gan14] Roman Ganhör. Athmos: Focus+context for browsing in mobile thumbnail collections. In *Proceedings of International Conference on Multimedia Retrieval*, ICMR '14, pages 65:65–65:72, New York, NY, USA, 2014. ACM.

[GI12] Ai Gomi and Takayuki Itoh. Mini: A 3d mobile image browser with multi-dimensional datasets. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, pages 989–996, New York, NY, USA, 2012. ACM.

[Han02] A. Hanjalic. Shot-boundary detection: unraveled and resolved. *IEEE Transactions on Circuits, Systems, and Video Technology*, 12(2):90–105, 2002.

[HCS14] Marco A. Hudelist, Claudiu Cobârzan, and Klaus Schoeffmann. Opencv performance measurements on mobile devices. In *Proceedings of International Conference on Multimedia Retrieval*, ICMR '14, pages 479:479–479:482, New York, NY, USA, 2014. ACM.

[HD12] Wolfgang Hürst and Dimitri Darzentas. History: A hierarchical storyboard interface design for video browsing on mobile devices. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, MUM '12, pages 17:1–17:4, New York, NY, USA, 2012. ACM.

[HGW07a] Wolfgang Hürst, Georg Götz, and Martina Welte. Interactive video browsing on mobile devices. In *Proceedings of the 15th international conference on Multimedia*, MULTIMEDIA '07, pages 247–256, New York, NY, USA, 2007. ACM.

[HGW07b] Wolfgang Hürst, Georg Götz, and Martina Welte. A new interface for video browsing on pdas. In *Proceedings of the 9th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '07, pages 367–369, New York, NY, USA, 2007. ACM.

[HH15] Wolfgang Hürst and Miklas Hoet. Sliders versus storyboards â€“ investigating interaction design for mobile video browsing. In Xiangjian He, Suhuai Luo, Dacheng Tao, Changsheng Xu, Jie Yang, and MuhammadAbul Hasan, editors, *MultiMedia Modeling*, volume 8936 of *Lecture Notes in Computer Science*, pages 123–134. Springer International Publishing, 2015.

[HIT86] David C Hoaglin, Boris Iglewicz, and John W Tukey. Performance of some resistant rules for outlier labeling. *Journal of the American Statistical Association*, 81(396):991–999, 1986.

[HL08] Mark J. Huiskes and Michael S. Lew. The mir flickr retrieval evaluation. In *MIR '08: Proceedings of the 2008 ACM International Conference on Multimedia Information Retrieval*, New York, NY, USA, 2008. ACM.

[HM08a] Wolfgang Hürst and Konrad Meier. Interfaces for timeline-based mobile video browsing. In *Proceedings of the 16th ACM International Conference on Multimedia*, MM '08, pages 469–478, New York, NY, USA, 2008. ACM.

[HM08b] Wolfgang Hürst and Philipp Merkle. One-handed mobile video browsing. In *Proceedings of the 1st International Conference on Designing Interactive User Experiences for TV and Video*, UXTV '08, pages 169–178, New York, NY, USA, 2008. ACM.

[HS88] S.G. Hart and L. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Human mental workload*, pages 139–183. P.A. Hancock and N. Meshkati (Eds.), Amsterdam: Elsevier, 1988.

[HSA14] Marco A Hudelist, Klaus Schoeffmann, and David Ahlström. Evaluating alternatives to the 2d grid interface for mobile image browsing. *International Journal of Semantic Computing*, 8(02):185–208, 2014.

[HSAL15] Marco A. Hudelist, Klaus Schoeffmann, David Ahlström, and Mathias Lux. How many, what, and why? visual media statistics on smartphones and tablets. In *Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, ICMEW '15. IEEE, 2015. to appear.

[HSB13a] Marco A. Hudelist, Klaus Schoeffmann, and Laszlo Boeszoermenyi. Mobile video browsing with a 3d filmstrip. In *Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval*, ICMR '13, pages 299–300, New York, NY, USA, 2013. ACM.

[HSB13b] Marco A. Hudelist, Klaus Schoeffmann, and Laszlo Boeszoermenyi. Mobile video browsing with the thumbbrowser. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 405–406, New York, NY, USA, 2013. ACM.

[HSL+10] Jochen Huber, Jürgen Steimle, Roman Lissermann, Simon Olberding, and Max Mühlhäuser. Wipe'n'watch: spatial interaction techniques for interrelated video collections on mobile devices. In *Proceedings of the 24th BCS Interaction Specialist Group Conference*, BCS '10, pages 423–427, Swinton, UK, UK, 2010. British Computer Society.

[HSST10] Wolfgang Hürst, Cees G.M. Snoek, Willem-Jan Spoel, and Mate Tomin. Keep moving!: Revisiting thumbnails for mobile video retrieval. In *Proceedings of the International Conference on Multimedia*, MM '10, pages 963–966, New York, NY, USA, 2010. ACM.

[HSX15a] Marco A. Hudelist, Klaus Schoeffmann, and Quing Xu. Improving interactive known-item search in video with the keyframe navigation tree. In *MultiMedia Modeling*, Lecture Notes in Computer Science. Springer International Publishing, 2015.

[HSX15b] Marco A. Hudelist, Klaus Schoeffmann, and Quing Xu. Multi-stripe video browser for tablets. In *MultiMedia Modeling*, Lecture Notes in Computer Science. Springer International Publishing, 2015.

[Hud12] Marco A. Hudelist. 3d image browsing for mobile devices, 2012.

[Hud13] Marco A. Hudelist. Next generation image and video browsing on mobile devices. In *Proceedings of the 3rd ACM Conference on International Conference on Multimedia Retrieval*, ICMR '13, pages 333–336, New York, NY, USA, 2013. ACM.

[IDC14a] IDC. Smartphone OS Market Share, Q3 2014. `http://www.idc.com/prodserv/smartphone-os-market-share.jsp`, 2014. [Online; accessed 5-Mar-2015].

[IDC14b] IDC. Smartphone Vendor Market Share, Q3 2014. `http://www.idc.com/prodserv/smartphone-market-share.jsp`, 2014. [Online; accessed 5-Mar-2015].

[IDC14c] IDC. Tablet OS Market Share 2014. http://www.idc.com/getdoc.jsp?containerId=prUS25267314, 2014. [Online; accessed 5-Mar-2015].

[IDC15] IDC. Global smartphone shipments forecast from 2010 to 2018. https://www.idc.com, 2015. [Online; accessed 19-Feb-2015].

[JDS08] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In Andrew Zisserman David Forsyth, Philip Torr, editor, *European Conference on Computer Vision*, volume I of *LNCS*, pages 304–317. Springer, oct 2008.

[Jur04] Steve Jurvetson. Geek bling bling - motorola razr v3. http://www.flickr.com/photos/44124348109@N01/1436702, 2004. [Online; accessed 4-Mar-2015 via Flickr; Creative Commons Attribution].

[Kau14] Bonifaz Kaufmann. *Handheld Projectors in the Wild - The Human Factors*. PhD thesis, Alpen-Adria Universität Klagenfurt, 2014.

[KB04] Amir Khella and Benjamin B. Bederson. Pocket photomesa: A zoomable image browser for pdas. In *Proceedings of the 3rd International Conference on Mobile and Ubiquitous Multimedia*, MUM '04, pages 19–24, New York, NY, USA, 2004. ACM.

[KJZ14] Kolbeinn Karlsson, Wei Jiang, and Dong-Qing Zhang. Mobile photo album management with multiscale timeline. In *Proceedings of the ACM International Conference on Multimedia*, MM '14, pages 1061–1064, New York, NY, USA, 2014. ACM.

[KKC12] Kwanghwi Kim, Sora Kim, and Hwan-Gue Cho. A compact photo browser for smartphone imaging system with content-sensitive overlapping layout. In *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, ICUIMC '12, pages 25:1–25:8, New York, NY, USA, 2012. ACM.

[KSFS05] Tim Kindberg, Mirjana Spasojevic, Rowanne Fleck, and Abigail Sellen. The ubiquitous camera: An in-depth study of camera phone use. *IEEE Pervasive Computing*, 4(2):42–50, April 2005.

[LCS11] S. Leutenegger, M. Chli, and R.Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2548–2555, 2011.

[Low04] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.

[LPE97] R. Lienhart, S. Pfeiffer, and W. Effelsberg. Video abstracting. *Communications of the ACM*, 40(12):54–62, 1997.

[LR13] Mathias Lux and Michael Riegler. Annotation of endoscopic videos on mobile devices: A bottom-up approach. In *Proceedings of the 4th ACM Multimedia Systems Conference*, MMSys '13, pages 141–145, New York, NY, USA, 2013. ACM.

[LXSS14] Xiaoxiao Luo, Qing Xu, Mateu Sbert, and Klaus Schoeffmann. F-divergences driven video key frame extraction. In *IEEE International Conference on Multimedia & Expo (ICME 2014)*. IEEE, 2014.

[MFF+09] Gregor Miller, Sidney Fels, Matthias Finke, Will Motz, Walker Eagleston, and Chris Eagleston. Minidiver: A novel mobile media playback interface for rich video content on an iphone. In *Proceedings of the 8th International Conference on Entertainment Computing (ICEC)*, volume 5709 of *Lecture Notes in Computer Science*, pages 98–109. Springer, Berlin / Heidelberg, Germany, September 2009.

[MKK11] Britta Meixner, Johannes Köstler, and Harald Kosch. A mobile player for interactive non-linear video. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 779–780, New York, NY, USA, 2011. ACM.

[MST00] Heimo Mueller-Seelich and Ed Tan. Visualizing the semantic structure of film and video, 2000.

[NW08] G. P. Nguyen and M. Worring. Interactive access to large image collections using similarity-based visualization. *J. Vis. Lang. Comput.*, 19(2):203–224, April 2008.

[PHBM09] Arto Puikkonen, Jonna Häkkilä, Rafael Ballagas, and Jani Mäntyjärvi. Practices in creating videos with mobile phones. In *Proc. MobileHCI*, pages 3:1–3:10. ACM Press, 2009.

[RBSW01] Kerry Rodden, Wojciech Basalaj, David Sinclair, and Kenneth Wood. Does organisation by similarity assist image browsing? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '01, pages 190–197, New York, NY, USA, 2001. ACM.

[RD06] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In Ale˘ s Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision - ECCV 2006*, volume 3951 of *Lecture Notes in Computer Science*, pages 430–443. Springer Berlin Heidelberg, 2006.

[RRKB11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2564–2571, 2011.

[RW03] Kerry Rodden and Kenneth R. Wood. How do people manage their digital photographs? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 409–416, New York, NY, USA, 2003. ACM.

[SA12a] K. Schoeffmann and D. Ahlstrom. Using a 3d cylindrical interface for image browsing to improve visual search performance. In *13th Int. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, pages 1–4, May 2012.

[SA12b] Klaus Schoeffmann and David Ahlström. An evaluation of color sorting for image browsing. *International Journal of Multimedia Data Engineering and Management (IJMDEM)*, 3(1):49–62, 2012.

[SAB11] Klaus Schoeffmann, David Ahlstrom, and Christian Beecks. 3d image browsing on mobile devices. *2013 IEEE International Symposium on Multimedia*, 0:335–336, 2011.

[SAB+14] Klaus Schoeffmann, David Ahlstrom, Werner Bailer, Claudiu Cobarzan, Frank Hopfgartner, Kevin McGuinness, Cathal Gurrin, Christian Frisson, Duy-Dinh Le, Manfred Del Fabro, Hongliang Bai, and Wolfgang Weiss. The video browser showdown: a live evaluation of interactive video search tools. *International Journal of Multimedia Information Retrieval*, 3:113–127, 2014.

[SAH14] K. Schoeffmann, D. Ahlstrom, and M.A. Hudelist. 3-d interfaces to improve the performance of visual known-item search. *IEEE Transactions on Multimedia*, 16(7):1942–1951, Nov 2014.

[SB12] Klaus Schoeffmann and Werner Bailer. Video browser showdown. *ACM SIGMultimedia Records*, 4(2):1–2, 2012.

[SC13] K. Schoeffmann and C. Cobarzan. An evaluation of interactive search with modern video players. In *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pages 1–4, July 2013.

[SCB14] Klaus Schoeffmann, Kevin Chromik, and Laszlo Böszörmenyi. Video navigation on tablets with multi-touch gestures. In *Proceedings of The Third International Workshop on Emerging Multimedia Systems and Applications (EMSA) at the IEEE International Conference on Multimedia & Expo (ICME 2014)*. IEEE, 7 2014.

[Sch10] Gerald Schaefer. A next generation browsing environment for large image repositories. *Multimedia Tools and Applications*, 47(1):105–120, 2010.

[Sch14] Klaus Schoeffmann. A user-centric media retrieval competition: The video browser showdown 2012-2014. *IEEE Multimedia Magazine*, pages 1–5, 2014. to appear.

[SFBJ08] Alan F. Smeaton, Colum Foley, Daragh Byrne, and Gareth J.F. Jones. ibingo mobile collaborative search. In *Proceedings of the 2008 International Conference on Content-based Image and Video Retrieval*, CIVR '08, pages 547–548, New York, NY, USA, 2008. ACM.

[SH98] Stephen J Sangwine and Robin EN Horne. *The colour image processing handbook*. Springer Science & Business Media, 1998.

[SHM+10] Klaus Schoeffmann, Frank Hopfgartner, Oge Marques, Laszlo Boeszoermenyi, and Joemon M. Jose. Video browsing interfaces and applications: a review. *Journal of Photonics for Energy*, pages 018004–018004–35, 2010.

[SOK06] Alan F. Smeaton, Paul Over, and Wessel Kraaij. Evaluation campaigns and trecvid. In *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, MIR '06, pages 321–330, New York, USA, 2006. ACM.

[ST94] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, Jun 1994.

[Sta14] Statista/IDC. Global market share held by tablet vendors from 2nd quarter 2011 to 4th quarter 2014. `http://www.statista.com/statistics/276635/market-share-held-by-tablet-vendors/`, 2014. [Online; accessed 5-Mar-2015].

[STB10] Klaus Schoeffmann, Mario Taschwer, and Laszlo Boeszoermenyi. The video explorer: A tool for navigation and searching within a single video based on fast content analysis. In *Proceedings of the First Annual ACM SIGMM Conference on Multimedia Systems*, MMSys '10, pages 247–258, New York, NY, USA, 2010. ACM.

[STF+12] G. Schaefer, M. Tallyn, D. Felton, D. Edmundson, and W. Plant. Intuitive mobile image browsing on a hexagonal lattice. In *Visual Communications and Image Processing (VCIP), 2012 IEEE*, pages 1–1, Nov 2012.

[Sui14a] Suite48Analytics. The Dispersed Photo Challenge Study. `http://www.suite48a.com/dispersed`, 2014. [Online; accessed 9-Dec-2014].

[Sui14b] Suite48Analytics. The Multi-Device Photo Use Study. `http://www.suite48a.com/multi-device-1/`, 2014. [Online; accessed 9-Dec-2014].

[SWS+00] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, Dec 2000.

[TFF08] A. Torralba, R. Fergus, and W.T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, Nov 2008.

[TGE11] Amrita Thakur, Michael Gormish, and Berna Erol. Mobile phones and information capture in the workplace. In *Ext. Abstracts CHI 2011*, pages 1513–1518. ACM Press, 2011.

[Tuk77] John W Tukey. Exploratory data analysis. *Addison-Wesley series in behavioral science: quantitative methods. 23 cm. 688 p*, 1977.

[WLW00] JamesZ. Wang, Jia Li, and Gio Wiederholdy. Simplicity: Semantics-sensitive integrated matching for picture libraries. In Robert Laurini, editor, *Advances in Visual Information Systems*, volume 1929 of *Lecture Notes in Computer Science*, pages 360–371. Springer Berlin Heidelberg, 2000.

[WSS+12] Marcel Worring, Paul Sajda, Simone Santini, David A. Shamma, Alan F. Smeaton, and Qiang Yang. Where is the user in multimedia retrieval? *IEEE MultiMedia*, 19(4):6–10, Oct 2012.

[XLL+14] Qing Xu, Yu Liu, Xiu Li, Zhen Yang, Jie Wang, Mateu Sbert, and Riccardo Scopigno. Browsing and exploration of video sequences: A new scheme for key

frame extraction and 3d visualization using entropy based jensen divergence. *Information Sciences*, 278(0):736 – 756, 2014.

[XLY+12] Qing Xu, Xiu Li, Zhen Yang, Jie Wang, Mateu Sbert, and Jianfu Li. Key frame selection based on jensen-rényi divergence. In *2012 21st International Conference on Pattern Recognition (ICPR)*, pages 1892 – 1895. IEEE, 2012.

[XPCL+10] Q. Xu, P.-Ch.Wang, B. Long, M. Sbert, M. Feixas, and R. Scopigno. Selection and 3d visualization of video key frames. In *Proceedings of IEEE International Conference on Systems Man and Cybernetics (SMC)*, pages 52 – 59, 2010.

[ZML+13] Jin-Kai Zhang, Cui-Xia Ma, Yong-Jin Liu, Qiu-Fang Fu, and Xiao-Lan Fu. Collaborative interaction for videos on mobile devices based on sketch gestures. *Journal of Computer Science and Technology*, 28(5):810–817, 2013.