TECHNISCHE
UNIVERSITÄT
DARMSTADT

# Quality-aware Content Adaptation in Digital Video Streaming

Am Fachbereich Informatik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertationsschrift

von

Stefan Wilk, M.Sc.
Geboren am 28. Januar 1987 in Zweibrücken

Erstreferent: Prof. Dr.-Ing. Wolfgang Effelsberg
Korreferent: Prof. Dr. Max Mühlhäuser
Korreferent: Prof. Roger Zimmermann (Ph.D.)

Tag der Einreichung: 13.10.2016
Tag der Disputation: 07.12.2016

Hochschulkennziffer D17
Darmstadt 2016

# ABSTRACT

User-generated video has attracted a lot of attention due to the success of Video Sharing Sites such as YouTube and Online Social Networks. Recently, a shift towards live consumption of these videos is observable. The content is captured and instantly shared over the Internet using smart mobile devices such as smartphones. Large-scale platforms arise such as YouTube.Live, YouNow or Facebook.Live which enable the smartphones of users to live-stream to the public. These platforms achieve the distribution of tens of thousands of low resolution videos to remote viewers in parallel.

Nonetheless, the providers are not capable to guarantee an efficient collection and distribution of high-quality video streams. As a result, the user experience is often degraded, and the needed infrastructure installments are huge. Efficient methods are required to cope with the increasing demand for these video streams; and an understanding is needed how to capture, process and distribute the videos to guarantee a high-quality experience for viewers.

This thesis addresses the quality awareness of user-generated videos by leveraging the concept of content adaptation. Two types of content adaptation, the adaptive video streaming and the video composition, are discussed in this thesis. Then, a novel approach for the given scenario of a live upload from mobile devices, the processing of video streams and their distribution is presented. This thesis demonstrates that content adaptation applied to each step of this scenario, ranging from the upload to the consumption, can significantly improve the quality for the viewer. At the same time, if content adaptation is planned wisely, the data traffic can be reduced while keeping the quality for the viewers high.

The first contribution of this thesis is a better understanding of the perceived quality in user-generated video and its influencing factors. Subjective studies are performed to understand what affects the human perception, leading to the first of their kind quality models. Developed quality models are used for the second contribution of this work: novel quality assessment algorithms. A unique attribute of these algorithms is the usage of multiple features from different sensors. Whereas classical video quality assessment algorithms focus on the visual information, the proposed algorithms reduce the runtime by an order of magnitude when using data from other sensors in video capturing devices. Still, the scalability for quality assessment is limited by executing algorithms on a single server. This is solved with the proposed placement and selection component. It allows the distribution of quality assessment tasks to mobile devices and thus increases the scalability of existing approaches by up to 33.71% when using the resources of only 15 mobile devices. These three contributions are required to provide a real-time understanding of the perceived quality of the video streams produced on mobile devices.

The upload of video streams is the fourth contribution of this work. It relies on content and mechanism adaptation. The thesis introduces the first prototypically evaluated adaptive video upload protocol (LiViU) which transcodes multiple video representations in real-time and copes with changing network conditions. In addition, a mechanism adaptation is integrated into LiViU to react to changing application scenarios such as streaming high-quality videos to remote viewers or distributing video with a minimal delay to close-by recipients.

A second type of content adaptation is discussed in the fifth contribution of this work. An automatic video composition application is presented which enables live composition

from multiple user-generated video streams. The proposed application is the first of its kind, allowing the in-time composition of high-quality video streams by inspecting the quality of individual video streams, recording locations and cinematographic rules.

As a last contribution, the content-aware adaptive distribution of video streams to mobile devices is introduced by the Video Adaptation Service (VAS). The VAS analyzes the video content streamed to understand which adaptations are most beneficial for a viewer. It maximizes the perceived quality for each video stream individually and at the same time tries to produce as little data traffic as possible - achieving data traffic reduction of more than 80%.

# KURZFASSUNG

Videoportale wie YouTube oder soziale Netzwerke wie Facebook verhalfen nutzergenerierten Videos zu einem enormen Erfolg. Zuletzt veränderte sich jedoch das Produktionsverhalten hin zu einer echtzeitnahen Verbreitung als Live-Videostrom. Dies ist möglich, da Mobilgeräte in der Lage sind aufgenommene Inhalte instantan an entfernte Betrachter zu verteilen. Heute müssen große Anbieter wie YouTube.Live, YouNow oder Facebook.Live jene Videoströme an zehntausende Betrachter gleichzeitig verteilen. Dies erreichen sie nur für geringe Bitraten und Auflösungen.

Jene Anbieter sind noch nicht in der Lage die Videoströme hochqualitativ und effizient zu sammeln, zu verarbeiten und zu verteilen. Ein Ergebnis hiervon ist eine vergleichsweise geringe, wahrgenommene Qualität. Effiziente Methoden werden erforderlich, da die aktuellen Kommunikationsnetze mit zunehmender Nutzung der Dienste überlastet sind.

In dieser Thesis wird das Konzept der Inhaltsadaption als eine Lösung zur effizienten und qualitätssensitiven Sammlung, Verarbeitung und Verteilung nutzergenerierter Videoströme diskutiert. Zwei Formen der Inhaltsadaption sind hierbei im Fokus der Arbeit: adaptives Videostreaming und Videokomposition. Beide Konzepte werden an verschiedenen Stellen des Videoproduktions- und Videoverteilungsprozesses angesiedelt um die wahrgenommene Qualität der Videoströme zu verbessern und den verursachten Datenverkehr zu verringern. Dabei werden die Adaptionen stets wohlüberlegt und geleitet durch die für den Betrachter eines Videostroms wahrnehmbare Qualität gesteuert.

Der erste Beitrag dieser Thesis ist ein besseres Verständnis was wahrgenommene Qualität bei nutzergenerierten Videos bedeutet und welche Faktoren diese beeinflussen. Hierfür werden in Nutzerstudien Qualitätsmodelle erstellt, die in dieser Form einzigartig sind. Jene Qualitätsmodelle erlauben den Entwurf neuartiger Qualitätsberechnungsalgorithmen. Gleichzeitig nutzen jene Algorithmen nicht ausschließlich visuelle Daten zur Berechnung, sondern vor allem Kontextinformationen. Dies ermglicht eine signifikante Beschleunigung der Verarbeitung und damit eine für Live-Videoströme notwendige echtzeitnahe Berechnung der Qualität ermöglicht. Der dritte Beitrag dieser Thesis ist eine Komponente zur Erhöhung der Skalierbarkeit der Qualitätsberechnung durch die Nutzung von stationären und mobilen Rechenkapazitäten. Jene Selektions- und Platzierungskomponente erlaubt die Bestimmung des besten Qualitätsberechnungsalgorithmus und dessen Ausführungslokation. Hierbei wird eine Erhöhung der Skalierbarkeit um bis zu einem Drittel erreicht, wenn die Ressourcen von 15 mobilen Endgeräten genutzt werden. Jene drei Beiträge erlauben die für die restliche Arbeit so wichtige echtzeitnahe Qualitätsberechnung.

Die Bereitstellung der Videoströme ist der vierte Beitrag dieser Arbeit. Jene Bereitstellung nutzt Inhaltsadaption um sich an verändernde Netzwerkbedingungen anzupassen und ferner Applikationsanforderungen oder Szenarienwechsel zu ermöglichen. Hierbei wird eine Form der Inhaltsadaption gewählt, die es auf Mobilgeräten ermöglicht verschiedene Repräsentationen desselben Videos zu enkodieren und in Echtzeit zwischen diesen zu wechseln. Hiermit kann zum einen eine echtzeitnahe Bereitstellung von Videoinhalten ermöglicht werden, aber auch ein hochqualitativer Videostrom mit größerer Latenz an entfernte Nutzer verteilt werden.

Die zweite Form und damit der fünfte Beitrag dieser Thesis ist die Videokomposition. Hierzu wird ein neuartiger, automatischer Videokompositionsalgorithmus für nutzergenerierte Live-Videoströme vorgeschlagen, der Kompositionsregeln von manueller Komposi-

tion lernt. Der vorgeschlagene Algorithmus ist der erste seiner Art, der echtzeitnahen Komposition einer Vielzahl an Videoströmen erlaubt und dabei Kriterien wie Videoqualität, -inhalt und kinematographische Regeln beachtet.

Der letzte Beitrag dieser Arbeit adressiert die Verteilung von Videoströmen durch den Einsatz einer inhaltsabhängigen, adaptiven Videoverteilung. Das vorgeschlagene System untersucht die zu verteilenden Videoinhalte hinsichtlich ihrer verschiedenen Repräsentationen und schlägt Adaptionen vor, die vorteilhaft für den Nutzer sind. Dabei zielt das System auf eine hohe, wahrgenommene Qualität bei gleichzeitiger Datenverkehrsreduktion ab. Das System erlaubt eine Datenverkehrsreduktion von über 80%.

# CONTENTS

# 1 INTRODUCTION

Digital video streaming places an enormous burden on existing communication networks. In the first six months of 2016, Internet-based video delivery of Netflix and YouTube accounted for 54.9% of North America's wired, downlink data traffic [**Sandvine2016**]. A shift is observable for users who not only consume, but increasingly distribute their own videos. Today, User-Generated Video (UGV) accounts for a huge proportion of daily data traffic - both download and upload - as YouTube accounts for 5.5% and Facebook for 19.09% of the upload traffic in wireless networks [**Sandvine2016**]. This trend is driven by the availability of smart mobile devices, e.g., smartphones. These devices include video cameras and a nearly ubiquitous access to the Internet for distributing captured videos. Their functionality and versatile capabilities are key to the idea of UGV, as anyone can capture videos anywhere at any time. Thus, mobile-generated data traffic is predicted to grow on average 54% per year until 2020, whereas the fixed network traffic grows by less than half (22% per year) [**Cisco2016**].

Similar to the trend of shifting from fixed to wireless networks, an observation can be made for the type of video streaming. It started in 2015 with Twitter's acquisition of Periscope[1] and Facebook's launch of Facebook.Live[2], and continued in 2016 with the relaunch of Google's YouTube.Live[3]. Video production behavior changed from recording, validating, and uploading to instantly broadcasting the recorded video as a live stream.

This thesis shows the challenges of reliably recording, uploading, and distributing user-generated live video streams. The aim of the thesis is to reduce the costs and increase the utility of user-generated live streams by proposing solutions to these challenges. The central concepts to achieve the goal are content adaptation and quality awareness. Content adaption describes the dynamic transformation of video to the requirements of the user's demands, device capabilities, and network conditions [**Rabin10**]. Quality awareness implies that content adaptation is performed in a way that maximizes a viewer's utility (perceived quality). In contrast to previous works [**Chen2015**, **Fu2015**, **Richerzhagen2016**, **Zhang2015**], this requires an end-to-end perspective - from the recording device to the playback on the receiving device.

This thesis looks at the challenging scenario of both video production and consumption of a video stream on a mobile device. Figure 1 illustrates the thesis' scenario, showing the upload of live video from a mobile device, its distribution through a fixed network, and the video's playback.

The arising challenges are many-fold, with the most difficult being the recording and distribution of high-quality content under the given resource constraints of recording devices and wireless networks. Recorded video streams are often of degraded quality, as the users lack professional hardware, such as tripods, for stabilizing a video, and recording skills. For assessing the perceived quality, the processing time is limited as the real-time characteristics of the produced video require instant transmission. This transmission is often realized using cellular networks, where a high throughput is only allowed for a capped

1 https://techcrunch.com/2015/03/13/how-periscope-works/; Visited on: 09/15/2016
2 https://techcrunch.com/2015/08/05/facescope/; Visited on: 09/15/2016
3 http://www.theverge.com/2016/6/23/12021232/youtube-launches-live-mobile-streaming-app; Visited on: 09/15/2016

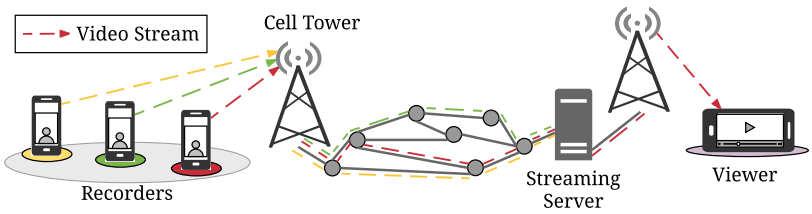



Figure 1: Overview of the live video upload scenario discussed in this thesis.

data volume per user. As soon as a data cap is reached, the available throughput is throttled to transmission speeds which do not allow high-quality video streaming. Also, the live upload of a video competes with other video streams and application traffic in the wireless networks they are connected to. These networks may have highly varying throughput rates and delays.

As the prediction of the network resources is uncertain and a single device has only limited capabilities to improve network conditions, the focus of this thesis lies on adaptation of the content - and not the network. This thesis proposes an efficient and quality-aware video collection and distribution service that leverages content adaptation.

## 1.1 RESEARCH CHALLENGES

The major research challenges in this thesis are listed as they influenced both the design decisions and the evaluation setup.

*Research Challenge 1: Understanding Quality in UGV*

Until recently, limited knowledge was available on the difference of professionally produced videos and user-generated ones, especially regarding the perceived quality. One difference between professional and amateur productions is that the lack of good recording equipment and skills induces further degradations to UGV. Models and algorithms are missing, which can detect and quantify the impact of these degradations on human perception. An understanding of the perceived quality is required to support quality awareness in content adaptation. Video inspection cannot be based on an in-depth analysis of the video itself, but must comply with the real-time constraints.

*Research Challenge 2: Wireless Communication Networks*

Devices considered in this thesis stream live video in wireless networks. What these technologies have in common is shared network access with other devices. In these networks, the available throughput for each device declines with increasing participant numbers. Similar to their demand for throughput, participating device numbers can rapidly change. As a result, each device has to cope with varying network conditions. Besides throughput constraints, different network technologies suffer from changing delays and have to cope with the mobility of the connected devices.

*Research Challenge 3: Real-time Constraints*

Live video implies strict timing constraints, which do not only apply to the transmission of the video but also for all associated tasks such as quality assessment. This constraint

limits the possibilities for in-depth inspection of a digital video, which makes sophisticated mechanism designs necessary.

*Research Challenge 4: Content Adaptation*

The last research challenge is the proposed content adaptation, which is realized in this thesis as a switching between varying quality versions of the same video (adaptive video streaming) and selecting the most appropriate time segments from different videos (video composition). Both concepts require an understanding of the perceived video quality. The effects of performing content adaptation on the perceived quality are unknown.

## 1.2 RESEARCH GOALS

From the described challenges, the research goals are derived. The main objective is the *design, realization, and evaluation of a live UGV uploading, processing, and distribution system leveraging quality-aware content adaptation*. Six subgoals are derived from this objective:

*Research Goal 1: Real-time quality assessment of UGV*

This thesis examines different influencing factors on the perceived quality in UGV. In an extensive study, the available algorithms are reviewed on their capabilities to reliably and in real-time predict the perceived quality of UGV. Quality models are derived from subjective studies, which build the basis for novel quality assessment algorithms proposed in this thesis.

*Research Goal 2: Content Adaptation: Investigating adaptive video streaming*

The second goal of this thesis is to investigate if adaptive video streaming can improve the efficiency as well as the achieved quality of digital video. Understanding the video content and the network conditions is required during the video streaming session.

*Research Goal 3: Content Adaptation: Investigating video composition*

The second form of content adaptation investigated in this thesis is the concept of video composition. We design a video composition application which dynamically selects the source of the next streamed video at a given time. The decision is made in a quality-aware manner. Similar to adaptive video streaming, video composition applications require a quality assessment of video streams in real-time.

*Research Goal 4: Network-efficient content adaptation*

Besides improved video quality, both forms of content adaptation shall address how minimal network costs for a user can be achieved while keeping the perceived video quality.

*Research Goal 5: Design of quality-aware video upload mechanisms*

Understanding the quality in UGV allows the design of an efficient upload mechanism, which extends existing work in the area. The upload mechanism has to cope with varying network conditions and mobility of the recording devices. Part of this mechanism is the investigation of content adaptation in the form of adaptive video streaming.

*Research Goal 6: Design of quality-aware, content-adaptive video distribution*

Finally, the video stream distribution to receivers is improved by investigating the potential for content adaptation when delivering video streams in a quality-aware manner. This distribution combines the results gained for real-time quality assessment, network efficiency, and content adaptation.

As a result, content-adaptive live UGV uploading, processing, and distribution were designed and realized, which were assessed according to their costs and utility. Two metrics measure the performance of the proposed contributions: the perceived quality of the delivered video streams (utility) and the generated data traffic (cost).

## 1.3 THESIS OUTLINE

This thesis is structured as follows. Chapter 2 elaborates on the fundamentals for digital video streaming, video broadcasting from a mobile device, perceived quality of UGV, and content adaptation. Besides an understanding of these fundamentals, the chapter offers insight into existing work, and gives a state-of-the-art survey on both quality-aware live UGV streams and content adaptation. A chapter summary shows missing links that are contributed by this thesis in the remaining chapters. In Chapter 3, quality models for UGV are presented. Derived from the related work on objective video quality metrics, a gap has been identified for the impact of recording degradations and recording positions. Chapter 4 presents novel quality assessment algorithms for detecting recording degradations and quantifying their impact on human perception. Furthermore, an approach for scalable processing of the algorithms is presented, which leverages not only the resources of a single server, but also the capabilities of smart mobile devices for efficient quality assessment. Chapter 5 introduces a novel, so-called Mobile Video Broadcasting Service (MBS), which allows a flexible and efficient upload of live UGV. Our approach abstracts from device mobility or network infrastructure and introduces adaptive video streaming into the domain of live video upload. Such an MBS is required for delivering the videos in time to the proposed video composition application in Chapter 6. It consists of two interchangeable approaches: A semi-automatic and an automatic video composition. The proposed video composition approaches show the first type of content adaptation. The second type of content adaptation is the adaptive video streaming, which is discussed in detail in Chapter 7. The proposed quality-aware video distribution offers a new way to ensure a high-quality video streaming experience and reduces the generated data traffic. An adaptive video streaming system is enhanced both by mechanisms for understanding the perceived quality on mobile devices, as well as by reliable content analysis and classification mechanisms. Chapter 8 concludes this thesis by giving a summary of the contributions and an outlook on further research directions.

# 2 BACKGROUND AND RELATED WORK

This chapter introduces the background on real-time quality assessment of UGV, its efficient upload from mobile devices, and its distribution. Furthermore, we discuss two types of content adaptation: adaptive video streaming and video composition. Afterwards we present a discussion of existing work on quality assessment of UGV in Section 2.3, the live video upload from mobile devices in Section 2.4, video composition in Section 2.5 and adaptive video streaming in Section 2.6. The following subsection highlights background information on digital video streaming and the understanding of the scenario discussed in this thesis.

## 2.1 DIGITAL VIDEO STREAMING

Digital video is the result of mapping real-world motion captured by a camera into the form of digital data. Video consists of digital images (*frames*), which are played back in sequence at a constant rate (*frame rate*). Independent frames consecutively played back are perceived as motion from around 14 to 18 Frames per Second (FPS) on [**Kandel2013**]. Even for cases in which rapid changes are captured, the individual frames are not seen as independent images but as a smooth motion as soon as the frame rate reaches 60 FPS [**Kandel2013**].

In this work, the focus lies on two-dimensional video, which maps the three-dimensional world onto a two-dimensional plane. Each of the video frames represents a raster image, in which each point contains visual information. These points are called *pixels*. The more pixels are available in a frame, the more visual information can be stored - usually allowing one to visualize more details. The number of pixels in a video is determined by its *resolution*, which is the width of a frame multiplied by its height in pixels.

Common video resolutions range from 768x576 (576p), which represents analog television, over high-definition resolutions 1280x720 (720p), 1920x1080 (1080p) up to 3840x2160 (2160p), also termed as ultra high definition. A common, minimum representation of each pixel requires a bit depth of 8 bits per channel, where recently the evolution to 10 bits per channel has begun [**Sullivan2012**]. It is evident that an encoding of each pixel in each video frame would cause an enormous size of a video. For example, for one second of 720p video at a frame rate of 30 FPS the resulting video would be of a size of approximately 79.1 MB.

Particularly for high resolutions, areas in a digital frame may have very similar structures and colors. Instead of describing these areas multiple times, a significant data size reduction can be achieved by describing redundant information only once. This step, which compresses the digital video is termed as video encoding. As this cannot only be applied to a single video frame, a compression can be extended to leverage redundancy between video frames. If areas in a single or different video frames are detected with similar characteristics, they need to be encoded only once. Note that for this thesis video encoding is solely discussed for compression reasons.

Prominent and widely applied video codecs - software or hardware which can compress and uncompress digital video - are H.264/Advanced Video Coding (AVC) [**Wiegand2003**] and H.265/High Efficient Video Coding (HEVC) [**Sullivan2012**]. These codecs determine

the level of compression and specify how many bits are required per second of digital video, the *bit rate*[1]. Besides many codec-specific features, they achieve this compression by quantization.

The concepts of frame rate, resolution, and bit rate are essential for the understanding of this thesis, as they describe the dimensions that allow adaptation[2].

Video streaming refers to the download of a digital video over a communication network and the in-parallel video playback before the download is completed. It is a delivery mode for digital video. A device capable of streaming video must be able to receive, decode, and render small parts of a video - so-called chunks. Also, it needs to ensure that the video chunks arrive in time. Thus, the network throughput should be equal or higher than the bit rate of the video. To compensate for small variations in the achieved throughput, streaming devices leverage the concept of a *playback buffer*, which stores video chunks before playback. Download rates higher than the video bit rate allow for filling the buffer faster than chunks being consumed by the video player.

Video streams can be distinguished by timing deadlines of the stream: Video on Demand (VoD) and live video [**Liu2003**]. Whereas VoD represents completely encoded video files - which can be requested and consumed at any time - the live video is distributed instantly while the content is being recorded and encoded. Well-known VoD services include the UGV platform YouTube[3], as well as the video streaming portals of Netflix[4] and Amazon Instant Video[5].

In contrast to VoD, live video streaming describes the real-time production, delivery and consumption of video. All these steps are realized during an event is happening. Environmental and legal conditions also affect what is perceived as a live video. For example, a live video broadcast from United States Television (TV) stations can be delayed between five seconds and five minutes[6], due to artificial delays invoked by stations to avoid fines for broadcasting explicit content. Still, in comparison to VoD the technical possibilities for distributing, caching and preloading of live video are rather limited.

## 2.2 APPLICATION SCENARIO

The scenario described in this thesis assumes an end-to-end transmission of a live video recorded on a mobile device using an access network technology, routing it through the core of a fixed network to a remote viewing device, which receives the stream (see Figure 2). In this scenario, the upload and distribution of a live video stream are decoupled by a server which receives the provided stream and prepares it for distribution.

The streaming server does not only consume a single video but can potentially process multiple, in-parallel recorded streams at the same time. It acts as a decoupling element of the video upload and its distribution. Preparation for the distribution on the server can be transcoding of different versions of a video for adaptive video streaming (see Section 2.2.2.1) or video composition (see Section 2.2.2.2).

1 In this thesis, *bit rate* describes a property of the video and not of the channel.
2 Recent codecs additionally allow to adapt the bit depth and the color gamut [**Sullivan2012**].
3 https://youtube.com; Visited on: 09/14/2016
4 https://netflix.com; Visited on: 09/14/2016
5 https://amazon.com; Visited on: 09/14/2016
6 http://news.bbc.co.uk/2/hi/entertainment/3478467.stm; Visited on: 09/14/2016

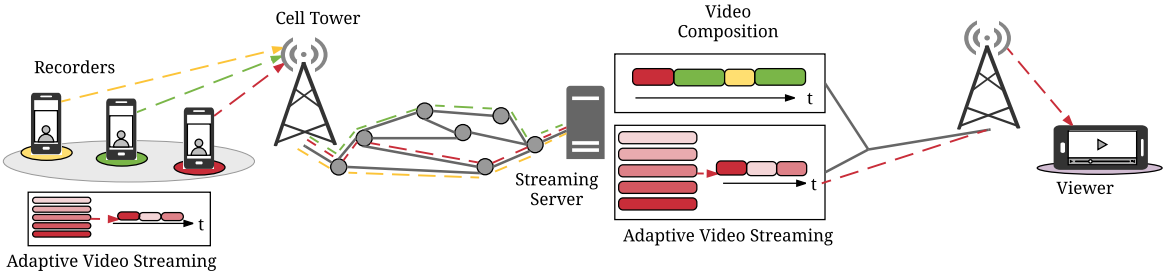



Figure 2: Options for content adaptation in the scenario introduced in Figure 1.

### 2.2.1 *Smart Mobile Devices*

The devices discussed in this thesis are assumed to be smart mobile devices. They incorporate four features: 1) they are capable of real-time capturing, encoding, decoding, and rendering of digital video, 2) they have additional sensors that describe the geographic position and capture the environmental conditions, i.e., the context, 3) they can be connected to at least one wireless communication network, 4) they are battery powered and can be freely moved.

#### 2.2.1.1 *Video Recording*

Smart mobile devices are also termed recording devices when their functionality of capturing a real-world scene as a digital video is in focus. Video recording is achieved by using the camera sensors in a recording device. In conjunction with a microphone, visual information and audio can be stored in a single digital video stream. This thesis focuses on the visual part of a digital video.

The recording device generates a two-dimensional representation of a scene. What a camera sensor captures is very much dependent on its technical capabilities: the technical design of the sensor, its resolution, angle of view, and focal length. For simplification reasons, the result of these different attributes is described as the *Field of View (FoV)* [**Ay2008**]. The FoV describes what is being captured in a scene and encoded in a digital video frame.

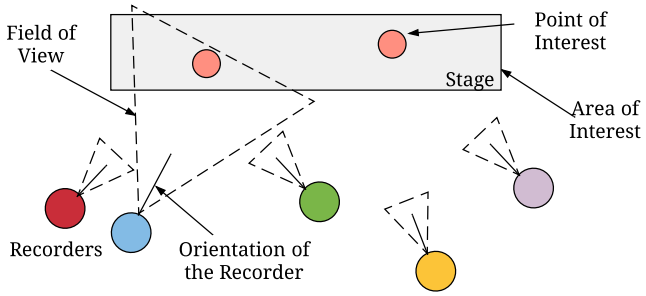



Figure 3: Illustration of the concepts RoI, AoI and FoV.

In a video frame, the *Region of Interest (RoI)* captures what is of specific interest for the recorder. The RoI represents a region in the video frame, which captures the visual and semantic information of a scene that is needed for understanding the captured content. Mapping the RoI of a recorded video to the real world allows retrieving coordinates, e.g., in the format of the Global Positioning System (GPS) as longitude, latitude, and altitude. For simplification reasons only the latitude and longitude values of a recorded scene are used to describe the *Area of Interest (AoI)*. Within an AoI different *Points of Interest (PoIs)*

can be positioned, which depict marks of interest, e.g., persons. The three concepts of FoV, RoI and AoI are illustrated in Figure 3. The concepts of FoV, RoI and AoI are not only required when different devices collaboratively record the same scene, but they as well as many algorithms proposed in this thesis can be applied to independent UGV streams.

#### 2.2.1.2 *Video Playback*

Each smart mobile device is capable of not only capturing live video but also receiving it. These devices leverage electronic circuit-based decoders for uncompressing digital video, and a media playback software (i.e., a video player) for rendering on the display. The playback experience is very much dependent on the context while watching a video stream. Section 2.6 discusses which influence the video stream, the encoded content and content adaptations have on the quality of a video playback.

#### 2.2.1.3 *Sensors*

To capture not only video and sound, smart mobile devices have also auxiliary sensors to understand the environment and context of a video recording or playback session. A reasonable classification of the different sensors available in today's smartphones is found in the documentation of the Android Operating System (OS). Available sensors are classified into motion, environmental, and position sensors[7].

*Motion sensors* measure forces applied to the smart mobile device, and differ depending on the type of force. What all of these sensors have in common is that they represent the forces in a tri-axial form (x,y,z). Accelerometers measure current acceleration forces in $\frac{m}{s^2}$ on each of the three physical axes. Gravity sensors are affected by the current gravity force without any device motion in all three physical axes ($[\frac{m}{s^2}]$). A gyroscope measures the rotation applied to a device in each of its axes in $\frac{rad}{s}$. An example of its use is to measure the orientation of the device.

*Environmental sensors* measure parameters to describe the context around a device. Examples of these sensors include ambient air temperature and pressure, humidity, and probably most famous the illumination sensor, which is used to adapt the screen brightness. A device's camera and microphone are also classified as environmental sensors.

Finally, *position sensors* describe the physical position of the device in longitude, latitude, and altitude. The most prominent example is GPS. The geographic orientation of a device is the fusion of the gravity sensor and a magnetometer to measure the geomagnetic field. Compass readings provided by some location providers are thus from a virtual sensor, which fuses gravity and magnetometer measurements.

#### 2.2.1.4 *Network Access*

Communication networks are used to allow different devices to communicate with each other. A communication network uses a single technology to interconnect autonomous devices to allow them to exchange data [**tanenbaum2003computer**]. A network can be part of other networks, so-called networks-of-networks, where the most prominent is the Internet [**tanenbaum2003computer**]. Here, the single technology used by all devices is the Internet Protocol (IP), which allows the addressing of devices and the routing of messages. Thus, IP offers a rich set of functionalities but by itself is not sufficient to enable communication between different devices. The concept of protocols reduces the complexity in

7 https://developer.android.com/guide/topics/sensors/sensors_overview.html; Visited on: 09/14/2016

communication networks, which encapsulate well-defined functionality. Protocols are classified to belong to specific layers in a protocol stack to illustrate how they communicate. Protocols on different layers do usually not know of each other. On the same layer, it is assumed that two communicating devices use the same or at least compatible protocols.

Communication networks can be further classified as wired or wireless communication networks. The latter use wireless connections between communicating devices. Usually, at least one participant of the wireless connection is a mobile device. This thesis follows the International Organization for Standardization (ISO)/Open Systems Interconnection Model (OSI) protocol stack (see [**tanenbaum2003computer**]). We mainly discuss innovations on the application layer of the stack. Due to the layered concept of the stack, the contributions of this thesis are not aware of the underlying network. IP and upper layer protocols hide the underlying technology from the application layer, making the proposed contributions independent of the used technology. A simple classification of wireless communication networks into cellular networks and Wireless Local Area Networks (WLANs) is used. For each of the categories, a prominent example is used, such as Long Term Evolution (LTE) [**Astely2009**] for cellular networks and Institute of Electrical and Electronics Engineers (IEEE) 802.11 [**tanenbaum2003computer**] for WLANs. For this thesis, it is assumed that two metrics can be easily calculated on the application layer for describing the performance of the communication network: the end-to-end application-layer *throughput* and *delay*. For this thesis, the *delay* describes the total time from sending a message until it is completely received. The *throughput* depicts the actual transmission speed of a channel. In contrast, we understand the bandwidth as a description of the theoretically, maximum amount of data that can be transmitted over a channel per time unit.

#### 2.2.1.5 *Mobility*

The mobility of the smart mobile devices induces challenges for both video recording and playback. First, to achieve mobility the devices are battery powered - this limits their maximum operating time. The investigation energy consumption of video recording, uploading and streaming is outside of the focus of this work.

Also, mobility affects the connection to a computer network, the reliability of individual hosts, the throughput, and the delay. In wireless networks, mobility can mean that the communication range of a stationary cell tower or access point is left so that connections are lost, and streaming or uploading of video is interrupted. An assumption in this thesis is that a detection of the device position and its mobility, i.e., at which speed it moves into which direction, is provided by a location provider on the smart mobile device.

### 2.2.2 *Content Adaptation*

The core of this thesis is the investigation of different types of content adaptation to achieve a reliable, high-quality streaming from mobile devices to mobile devices. Content adaptation is a concept already used in the web for the delivery of websites. It is the process to transform a page based on the capabilities of the device or a communication network, as well as to the user and his experience or preferences [**Rabin10**].

Similar concepts are proposed for the creation, transmission and consumption of live video streams. In this work, two types of content adaptation are discussed: the adaptive video streaming and video composition. Both concepts can be placed on the recorders, the servers and the receiving video streaming clients (see Figure 2). The conceptual difference of the two content adaptation types is illustrated in Figure 4. Whereas adaptive video

streaming provides adaptation within a video, video composition leverages adaptation between videos.

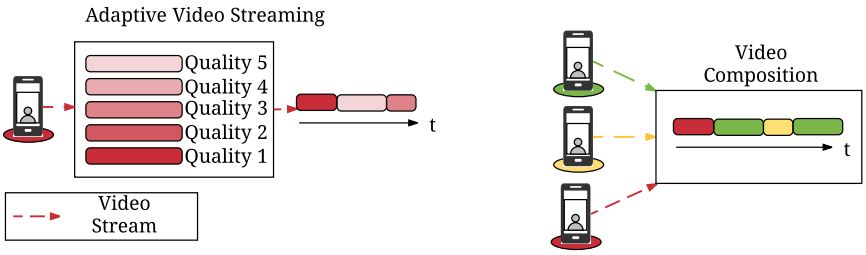



Figure 4: Conceptual difference of adaptive video streaming and video composition.

#### 2.2.2.1 *Adaptive Video Streaming*

Adaptive video streaming is usually applied to the delivery of a digital video to compensate variations in the network conditions, mostly the throughput, during playback [**DeCicco2010**]. A digital video is made available in different versions, where each version is different regarding the average bit rate. The video could also change in the other dimensions such as the frame rate or the resolution.

Adaptive video streaming allows devices to switch between different versions of a video, e.g., for adjusting the video stream bit rate to network conditions. Adaptive video streaming can be supported by the video encoding. Two categories of video encodings are discussed: the Single Video Layer enCoding (SVLC) and Multiple Video Layer enCoding (MVLC).

*Single Video Layer enCoding (SVLC)*

The advantage of SVLC is the wide support regarding hardware encoders and decoders and software video players. When using SVLC each video version is a single, independent decodable video file. Currently, all of the supported video encodings on mobile phones are based on SVLC, such as H.264/AVC, VP8, H.265/HEVC or VP9 [**Feller2011**, **Mukherjee2013**, **Sullivan2012**, **Wiegand2003**].

*Multiple Video Layer enCoding (MVLC)*

MVLC supports adaptive video streaming by organizing the video representations as interdependent layers. Each layer encodes the delta of information to the next lower layer. It reduces the redundancy between video representations to a minimum. Additional data is needed to organize the interdependencies. The main advantage of using MVLC for many streaming applications is solely a reduced storage requirement on the streaming server.

Some of the latest SVLC approaches have multi-layered variants, such as H.264/Scalable Video Coding (SVC) [**Schwarz2007**] and H.265/Scalable High Efficient Video Coding (SHVC) [**Boyce2016**]. Besides its complexity in decoding, it has been shown that the imagined benefits are very limited in practice. The overhead using MVLC is higher than the saved data traffic in comparison to SVLC, when more than three layers are encoded [**Grafl2013**, **Wang2013**].

Note that multiple description coding is a related approach to MVLC which has been omitted due to lack of support on today's mobile or stationary devices [**Setton2008**]. Details on the current state of adaptive video streaming systems are given in Section 2.6.

#### 2.2.2.2 *Video Composition*

Whereas adaptive video streaming leverages different versions of the same content, video composition combines segments of different videos to create a new video stream. In the case of a given application scenario, the content that can be selected is represented by the recording devices. Video composition can select which view to show at a given time from all the available video recording sources.

The concept of video composition, as well as influencing factors like its effect on the perceived quality, are discussed in Section 2.5.

## 2.3 PERCEIVED QUALITY OF UGV

A central aspect of content-adaptive media streaming is to gain a detailed understanding of the perceived quality for the viewer. Quality is defined as the "[...] evaluated excellence or goodness [...]" or "[...] the degree of need fulfillment [...]" [**Qualinet2013**]. The focus lies on the discussion of quality as it is perceived for UGV, as well as existing methods to determine it.

### 2.3.1 *Overview of the Video Streaming Process*

The process of UGV streaming, from the capturing of a video until the playback on a mobile device, is an extension of the recording steps proposed by Jang et al. [**Jang2016**]. To determine the perceived quality, influencing factors from all process steps (see Figure 5) need to be considered.

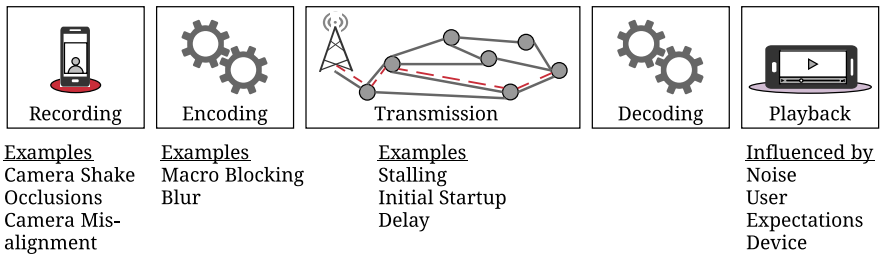



Figure 5: Video streaming process and its problems as understood in this thesis.

It starts on the video recording side, where the users' actions influence the perceived quality. The process continues over the compression of the video (encoding) and the transfer over a communication network such as the Internet. It ends at the viewer's side with the decoding and rendering of the video on a display.

#### 2.3.1.1 *Recording Step*

The first step is the recording of a real-world scene by a non-professional cameraman. In this thesis, it is assumed that the recording device is a smart mobile device as discussed earlier.

There are two essential differences between professional productions and UGV: 1) the skills of recording users and 2) the equipment used for capturing a video. Professional equipment consists of cameras, which include physical or digital stabilization mechanisms to create stable video frames. These mechanisms allow a controlled focus, panning and tilting of the camera without any disturbing (unintended) motion to occur. Professional

cameras are mounted on tripods, dollies or cranes to support a broad range of applications. Further optical and digital mechanisms help to capture videos, e.g., adjustable lenses to capture different distances.

In contrast to professional cameras, today's smartphones have cheap lenses with large apertures causing degraded quality in low light settings [**Dialogic2009**]. Also, sensors need to comply to low energy consumption requirements, making recorded frames less attractive. As a result, the recorded videos are not of the same quality as if produced by professional cameras.

Also, most degrading effects occur due to the lacking skill of users. Especially the degradations caused by camera shakes, harmful occlusions and camera misalignments are to be mentioned. *Camera shake* is caused by uncontrolled movements of the recording user due to the lack of a stabilizing tripod. If such movements occur continuously with varying directions, they are named camera shakes. In contrast to camera shakes, intended motions of the camera include tilting and panning. Tilting describes motion about the horizontal axis of the camera, whereas panning occurs about the vertical axes. In contrast to shake, these intended forms of motion do not include repeated direction changes [**Ward2003**]. Some recent high-end smartphones compensate unintended motion using optical image stabilization[8]. An extra gyroscope is used to control movements of an adjustable camera to compensate for unintended movements. It works in the range of tenths of millimeters for each exposure [**Karpenko2011**, **Shin2011**].

*Harmful occlusions*, also termed as occultations, depict that a foreground object blocks the view of a background object, which is in the RoI of a video frame. The RoI illustrates that such an occlusion is perceived as distracting if the viewer is interested in watching the background object. Other occlusions may not be perceived as distracting as they are a natural part of a scene. A harmful occlusion often occurs in UGV as users move while recording, and objects cross the line of sight of a recording device.

*Camera misalignments* represent the case when smart mobile devices do not capture the commonly agreed AoI, e.g., the stage during a concert. Misalignments start from slightly drifting away from the RoI [**Bowen2013**] to its worst form, where the recording does not capture the RoI at all. A second type of misalignment addresses the orientation along the axis describing the device's viewing direction (z-axis). Users may change from a perfectly aligned orientation to slightly tilted recordings. This tilt is measured as the difference of the recorded plane (consisting of horizontal and vertical axes) to the reference plane of a scene.

#### 2.3.1.2 *Encoding*

Typical compression algorithms lead to a lossy conversion of the input, which results in information loss that cannot be compensated on the decoding side. As discussed earlier in this chapter, compression is achieved by leveraging the redundancy available in and between video frames. Common encoding degradations address both spatial effects within a single frame and impairments reducing the perception of motion in the video (see [**Jang2016**, **TaoLiu2010**] for an overview on degradations). Encoding is a very resource-consuming process, which does not always achieve satisfying results on smart mobile devices. If the resource demand is too high, this usually results in the skipping of captured frames before encoding - thus reducing the possible frame rate. Stohr et al. conducted work illustrating this effect for the live streaming platform YouNow [**Stohr2015**]. The authors show that the average and for most technologies, the highest frame rate observed is

8 http://www.cultofmac.com/390139/optical-image-stabilization-iphone/; Visited on: 09/15/2016

still lower than what the average human perceives as smooth motion [**Kandel2013**]. The resulting video streams are perceived as jerky (motion jerkiness), which degrades the perceived quality.

#### 2.3.1.3 *Transmission*

Degrading effects can also occur during the transmission of a video stream over a network. Degradations consist of packet losses that result in a degraded video decoding, the initial startup delay, and freezing playback for rebuffering (stalling) effects.

In best effort delivery schemes, sent packets of a media stream can be lost or delayed so that the video receiving side cannot completely decode the video [**Dialogic2009**]. Such a packet loss may affect an entire frame or only a part of its data [**TaoLiu2010**] and existing video encoding standards implement methods to compensate a small percentage of packet losses without significantly degrading the viewing experience [**Sullivan2012**, **Wiegand2003**].

Besides that, user experience can be degraded due to the initial waiting phase for the video stream or frequent interruptions during playback, i.e., stalling [**Hossfeld2012**]. Both happen when network capacity is lower than the video bit rate, which implies that an in-time delivery of the video stream is not possible. Simply said, the video is being consumed faster than it is delivered. Recent studies show that stalling significantly degrades the viewing experience [**Hossfeld2012**, **Hossfeld2013**, **Hossfeld2014**].

#### 2.3.1.4 *Playback Context*

User perception, device capabilities, and environmental conditions affect how a video stream is perceived at the receiver's side.

User perception is composed of characteristics which can be any (in)variant property of the person, which describes its demographic and socio-economic background, current emotions, or physical as well as mental constitution [**Qualinet2013**]. Such characteristics can be user's viewing habits as well as disabilities such as color blindness.

The device can influence the perception by any of its features that determine the technically produced and measurable quality of the video stream [**Qualinet2013**]. In the context of video playback, this addresses factors such as the decoding quality, display size, and brightness, as well as available decoders on the device.

Finally, the environmental conditions and the context describe any property of the physical, temporal, and social context of a video playback session [**Qualinet2013**]. It contains lighting conditions, ambient noise or any other distraction, as well as the time of day and cost of a service.

The integration of the context during video playback for the perceived quality has attracted research initiatives that have not yet been widely accepted [**Kroupi2014**, **Luo2008**, **Moorthy2012**, **Scholler2012**]. This area is not in the focus of this work, but the effects of using mobile devices with reduced display sizes for video playback are considered in Chapter 7.

### 2.3.2 *Subjective Quality Assessment*

The most reliable method to determine the quality of a video streaming session is to ask the viewers of the stream. This method can be reproduced as large-scale subjective experiments, where multiple subjects assess the video stream's quality regarding standardized

assessment scales [**Winkler2008**]. Another advantage of subjective quality studies is that it can be applied to degradations occurring at any step of the video streaming process (see Section 2.3.1 and Figure 5).

For performing a subjective quality study, multiple subjects are used to mitigate subjective preferences and retrieve a mean opinion on the quality. The subjects conduct their judgments on standardized quality assessment scales. Following the rules of the International Telecommunication Union (ITU), the subjective experiments are conducted under controlled conditions [**ITU-R2012**, **Winkler2009**]. Quality assessment scales discussed in this thesis include the ITU recommendation Single Stimulus Continuous Quality Scale (SSCQS) [**ITU-R2012**].

Judgments are then aggregated into subjective quality metrics. In this thesis, the metrics Mean Opinion Score (MOS) [**ITU-J800**] and Just Noticeable Difference (JND) [**Watson2001**] are used. Conducted subjective experiments and resulting quality metrics build the benchmark for all objective quality metrics [**Winkler2008**]. For the interested reader, further subjective metrics are introduced by Hossfeld et al. [**Hossfeld2016**], and quality assessment scales are discussed by the respective ITU recommendations [**ITU-R2012**, **ITU-J800**].

#### 2.3.2.1 *Assessment Scales*

Subjective experiments conducted in the thesis ask users to rate the quality of video sequences one-by-one, or select the highest-quality sequence between a reference video and an impaired sequence.

Evaluations that assess the absolute quality of a video sequence individually leverage the SSCQS, which ranges from one to five. Here, one represents the worst quality imaginable (bad) and five the highest possible quality (excellent) [**ITU-R2012**]. In a User Interface (UI), the scale is usually represented as a slider, where each integer values is annotated by the respective quality descriptions - 1: bad, 2: poor, 3: fair, 4: good and 5: excellent (see Figure 6).

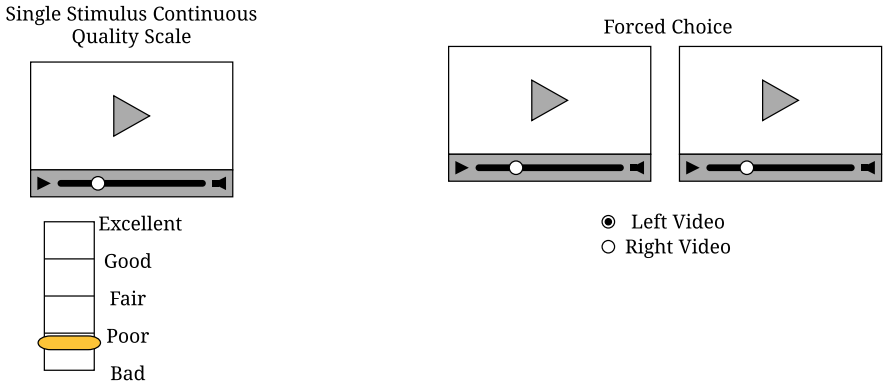



Figure 6: Subjective quality assessment scales: Single Stimulus Continuous Quality Scale (SSCQS) and Forced Choice Experiment.

This evaluation methodology is suitable to determine quality as perceived in real streaming sessions, as the viewer does not know in advance if a degradation is present or not. The SSCQS is the basis for the subjective quality metric MOS.

A forced choice experiment (see Figure 6) is set up when it should be determined if a degradation can be detected in a direct comparison of two video sequences. Instead of a continuous scale, the viewer decides which video sequence has the higher quality. This binary decision is the basis for the JND [**Keelan2003**].

#### 2.3.2.2 *Metrics*

As mentioned above, MOS is a concept to describe the subjective perception of quality, and is calculated from the SSCQS [**Suarez2016**]. The MOS of a video ranges from 1 (bad) to 5 (excellent). The MOS is calculated as the average of the subjects ratings as

$$MOS = \frac{1}{N}\sum_{i=1}^{N} r_{ij} \tag{1}$$

where $N$ represents the number of assessments for a video $j$ and $i$ is the index of a subject. Thus, $r_{ij}$ represents a quality rating of one user for one video.

As ratings of individuals may be very diverse, a data cleaning and a rating normalization task are performed. The data cleaning is recommended by the ITU-R BT.500 and identifies inconsistently rating subjects. The procedure is discussed in the Annex B of ITU-R BT.500 in Section 2.3.2 [**ITU-R2012**].

For the resulting ratings, a normalization is conducted. It ensures that subjective assessments are comparable and is proposed by Simone et al. and refined by Horch et al. [**DeSimone2010**, **qualitycrowd**].

For normalizing the ratings we calculate the value $r'_{ij}$, which is derived from the original rating $r_{ij}$ corrected by the average of ratings of the assessor $\overline{r_i}$. Also, the mean across all assessors and video sequences $\overline{r}$ is used for normalization:

$$r'_{ij} = r_{ij} - (\overline{r_i} - \overline{r}) \tag{2}$$

The MOS aggregates all ratings by:

$$MOS_j = r_j = \frac{\sum_{i=1}^{N} r'_{ij}}{N} \tag{3}$$

A second quality metric is the JND, which is determined in a forced choice experiment [**Keelan2003**, **Watson2001**]. The JND is an ISO standard for the assessment of quality in images and videos. This method tries to determine the minimum difference $\delta_{\Psi_{A,B}} = \Psi_A - \Psi_B$ between two video signals A and B, which is noticeable by a human. $\Psi_A$ represents the perceived quality of the video A and $\Psi_B$ the quality of video B. The JND can be determined using forced-choice tests in which each subject rates the same two video sequences - one being an impaired sequence and the other an unimpaired video. The viewer does not know which of the two video sequences is unimpaired. The order of comparisons is randomly selected and repeated several times for different subjects. In each evaluation round, the subject has to decide which stimulus is the best. If a stimulus receives 75% of the votes, it achieves a JND of one [**Thang2014**]. The generalized model is that the $\delta_{\Psi_{A,B}}$ is represented in JND units as $\delta_{\Psi_{A,B}} = \frac{12}{\pi} * arcsin(\sqrt{v}) - 3$, where $v$ is the fraction of votes ([0,1]) for one video [**Thang2014**].

### 2.3.3 *Objective Quality Assessment*

Objective quality assessment predicts the quality an average user would perceive when watching a video sequence [**Winkler2008**]. The algorithms can be divided into Full Reference (FR), No Reference (NR) and Reduced Reference (RR) metrics. Video quality metrics belonging to the FR category require a perfectly preserved reference video sequence, which is often used in conjunction with the potentially impaired test sequence [**Winkler2008**]. FR metrics detect the physical differences (pixel differences) in the two sequences easily. The

challenge is to map the detected differences to human perception to understand their impact on the perceived quality. Since every pixel of every video frame of a sequence is compared with its reference, the processing time and required computational resources can be enormous. The practical usage of FR metrics in streaming scenarios is often limited as the delivery of the unimpaired video sequence is inefficient or unrealizable.

NR video quality assessment examines solely the test video without any need for a reference [**Winkler2008**]. The accessibility of the original video may be impractical or problematic because the output of a camera may already be compressed. NR quality assessment approaches are more suitable to be used for assessing video streaming scenarios as only a single video is needed for assessment. At the same time, these metrics achieve a significantly reduced correlation with subjective quality assessments in comparison to FR approaches [**Winkler2009**].

RR metrics describe an intermediate approach, which does not require a reference video sequence, but solely some metric-specific features which are extracted from the reference video. The RR metric uses the test sequence and the extracted reference features for the quality prediction.

In the remaining section, existing objective quality metrics are described and compared. This discussion is split into the assessment of user-generated degradations and the analysis of standardized algorithms which detect impairments induced by the compression, the transmission or the playback of the video stream.

This distinction is chosen, as the set of standardized and evaluated algorithms focuses on the effects of compression and transmission effects on the video, thus, excluding degradations occurring while recording.

#### 2.3.3.1 *Recording Quality Assessment*

Degradations caused by the user's actions are termed as recording degradations. As mentioned above in this thesis, some of the most severe degradations are discussed: camera shakes, harmful occlusions and camera misalignments. For an extensive discussion of other degradations in UGV, please refer to the work of Jang et al. [**Jang2016**].

It is important to know that only NR metrics can be leveraged, as no reference video is available. The algorithms discussed for camera shake, harmful occlusion and camera misalignment assessment are thus NR metrics.

##### *Camera Shake Assessment*

Camera shake assessment on mobile recording devices is achieved by either analyzing the video or auxiliary sensor samples gathered during a recording session. Video-based algorithms are the most prominent ones; they achieve a high reliability at the costs of a high runtime. An early approach towards the detection of camera shakes is proposed by Campanella et al. in their work targeting at summarizing home videos automatically [**Campanella2007**]. The approach leverages the Luminance Projection Correlation (LPC) algorithm on each video frame and compares consecutive video frames to detect occurring panning and tilting, so the horizontal and vertical movements of a camera [**Nagasaka1999**, **Uehara2004**]. The consecutive frames are put into fixed size segments to be analyzed. A threshold is used to filter only significant motion that is then classified as a shake using the following equation:

$$S = \frac{1}{N} \sum_{i=1}^{N} \sqrt{(pan_i - fpan_i)^2 + (tilt_i - ftilt_i)^2} \tag{4}$$

Here, N represents the number of frames in a video segment, $pan_i$ and $tilt_i$ are original pan and tilt values, $fpan_i$ and $ftilt_i$ are low-pass-filtered pan and tilt values. Here, $i$ represents the current frame index. The quality impact of the detected shakes in a video segment is then determined as $Q_{CM} = \frac{S_{max}-S}{S_{max}}$. Values for $S_{max}$ are not given by Campanella et al. - limiting the reproducibility of results [**Campanella2007**].

The approach of Campanella et al. has attracted interest in the research community [**Campanella2007**]. Many of the video composition applications leverage (and slightly extend) the approach [**Bano2015b**, **Saini2012**, **Shrestha2010**]. Saini et al. replace the low-pass filter preprocessing step by a median filter to better distinguish intended camera motion, i.e., pan and tilt, from camera shakes [**Saini2012**]. Also, the final shake score is the sum of absolute differences of the original motion vectors and median filtered motion vectors, and follows Equation 4. The score calculation is applied to a window of 100 frames and post-processed in order to normalize the values to the range of [0,1].

Abdollahian et al. introduce a machine learning approach for the classification of camera motion [**Abdollahian2010**]. On the basis of detected panning and tilting using a pixel matching approach called the Integral Template Matching, a Support Vector Machine (SVM) is trained to classify shakes, blur, stability, and zooms [**Dong-JunLan2003**]. On the given dataset, a classification rate of around 87.32% is reported - but no relation to the perceived quality is given.

On the other side, auxiliary sensor-based algorithms are proposed for camera shake detection. Cricri et al. propose to leverage the compass on smartphones and applying a low-pass filter to identify camera pan, tilt and shakes [**Cricri2012**]. The algorithm maps the real-time gathered compass sensor readings to the video. Next, motion detected by the compass is classified in three steps: 1) low-pass filtering of raw compass data, 2) computation of the first discrete derivative on the compass readings, and 3) a peak detection based on a predefined threshold. Camera shake can be detected and removed in the first step by applying the low-pass filtering. This approach only works for very short shaky segments of a video.

This work was extended by investigating the accelerometer as a source for reliably detecting camera shakes [**Cricri2012**]. The underlying assumption is that frequency contributions between 10 to 20 Hz are caused by camera shakes [**Yu2007**]. The resulting algorithm leverages a high-pass filter at a frequency of 10 Hz in all three axes of the sensor (y,y,z). On the basis of the filtered values for each axis, the variance is calculated as $\sigma^2 x$, $\sigma^2 y$ and $\sigma^2 z$. The median of the three values is chosen in order to avoid outliers on a single axis to impact the shake intensity measurement.

Bano et al. propose another auxiliary sensor approach, leveraging the gyroscope of smartphones [**Bano2015**]. They aim at dissecting pan and tilt from the camera shakes. In a first step, the radial component of a gyroscope sensing is computed as $G_r(t) = \sqrt{G_x(t)^2 + G_y(t)^2}$, where $G_x(t)$ represents the degree of motion sensed on the x-axis, and $G_y(t)$ the proportion on the y-axis. A shake is then detected and stored as a binary variable $S(t)$ as

$$S(t) = \begin{cases} 1 & \text{if } G_r(t) > \beta, \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$\beta = 0.06$ is an empirically determined shake threshold.

Liu et al. [**Liu2014**] propose an algorithm to classify camera motion into intended motion or unwanted motion. They leverage similar, previously mentioned models by using the accelerometer as a source. Similar to other approaches, these are either video- or auxiliary sensor-based, and no mapping to the perceived quality is given.

*Harmful Occlusion Detection*

Occlusion detection is a required step in object or template tracking [**Koller1994**, **Nguyen2001**, **Saravanakumar2012**, **Weng2006**]. Weng et al. present one example of this category [**Weng2006**]. The proposed algorithm can track moving objects even in the presence of occlusions. An object is tracked after it is annotated manually. To track the object, a region growing segmentation approach is applied in the frames $t$, $t-1$, $t+1$. Similarly, a segmentation is performed on the dominant colors of the respective video frames. An occlusion of an object is detected when the ratio of the object's area in the frame $t$ compared to $t-1$ shrinks, and similarly, its size decreases from frame $t$ to $t+1$. Those models try to detect occlusions but give no reasoning on the impact of the occlusion on the quality.

Saini et al. propose the most promising approach when it comes to a low computational complexity and an application to UGV [**Saini2012**]. The underlying assumption is that occluding objects must be closer to the camera and thus show a lower edge density than the occluded objects, which exhibit a higher distance to the recording camera. By continuously checking for significant differences in the edge densities in a scene, candidates for occlusions can be detected. Each video frame is first represented in an edge representation using the Canny Edge Detector [**Canny1986**]. A binary representation of a video frame is derived, which depicts an edge pixel as 1 and a non-edge pixel as 0. A convolution matrix is applied to the binary edge representation using a unity kernel matrix $W$ (usually a matrix of ones): $I^d = I^e \odot W$ where $\odot$ represents the convolution operation. As harmful occlusions occur at specific regions and seldom occlude the whole video frame, subblocks of the frame are constructed, where each block has a size of $b \times c$. For each block, it is determined if it is occluded by calculating the sum of the edge densities and determining if it is less than the threshold $T^e$. The authors state that $T^e$ should represent a meaningful, empirically determined value for separating foreground from background objects. As neighboring blocks could indicate diverging statements on whether the whole frame region is occluded or not. Then a connected component analysis is conducted, which tries to find the largest group of connected (occluded or non-occluded) blocks in a frame. Gaps are labeled as occluded if the majority of their neighboring blocks are occluded. An occlusion score $S^O$ is computed by the fractional occluded region $f$ as

$$f = \frac{N_O}{N_{Total}} \tag{6}$$

where the resulting occlusion score is represented as $S^O = 1 - e^{-f}$. $N_O$ represents the blocks labeled as occluded and $N_{Total}$ the total number of blocks. The authors state, that for a block size of $20 * 15$ pixels an occlusion score of more than 0.2 can be classified as disturbing. Further information on the relation of the algorithm to the perceived quality is not given.

*Camera Misalignment and Orientation*

Similar to the camera shake assessment, the misalignment detection can be classified into algorithms either inspecting the video or leveraging auxiliary sensor data. An additional criterion is the detection of either misaligned recordings for a given RoI or the issue of tilting the camera during the recording. Most of the existing algorithms focus on camera orientation, but none quantifies the impact of a view not being perfectly aligned with the RoI [**Cricri2014**, **Saini2012**].

Saini et al. propose a video-based algorithm for camera orientation detection relying on the distinction of horizontal to non-horizontal edges [**Saini2012**]. It is assumed that

the majority of edges are horizontally aligned, when a smart mobile device records in landscape mode. Camera orientation is thus defined as a rotation of the camera around the horizontal axis of the recording device. From an edge representation of the video frames, a *Hough transform* is performed in order to detect straight lines, where $l_i$ represents the length of the $i^{th}$ line and $o_i$ is the angle in relation to the horizontal plane. It is assumed that a camera is tilted by less than $\pm\frac{\pi}{4}$, corresponding to $\pm 45°$. Thus, any higher angle describes that the respective line is noise. The resulting orientation is $o_i$ for a line $i$, which is used for determining the camera orientation as:

$$CO = \frac{\left| \frac{1}{N^l} \sum_{i=1}^{N^l} o_i * l_i \right|}{\pi/4} \tag{7}$$

Here, the score is obtained on the basis of the absolute mean orientation being weighted and normalized by $\frac{\pi}{4}$ (45°).

Cricri et al. propose a method for orientation detection by making use of auxiliary sensors (accelerometer or compass) of smartphones [**Cricri2011**]. The wrong orientation is determined by analyzing the static component of accelerometer readings without any motion of the device itself. For each intended display orientation, either landscape or portrait, a contribution to the accelerometer axis can be determined. In a properly oriented camera, one axis should not sense any contribution to another axis, e.g., on the y-axis in the landscape orientation. The approach of Cricri et al. leverages the low frequency accelerometer proportions by preprocessing the data using a low-pass filter at 10 Hz. The resulting filtered readings are used to combine the instantaneous orientations $O_I$ as

$$O_I = \arctan \frac{A_y}{\sqrt{(A_x)^2 + (A_z)^2}} \tag{8}$$

If $O_I$ is larger than a predefined threshold $T_{O_I}$ - which represents the acceptable deviation angle - the recording is classified as the wrong camera orientation.

*Discussion*

Except for the camera shake assessment, only a limited set of algorithms exist for detecting severe recording degradations. All algorithms suffer from a lack of validated quality models. Existing work does not quantify the perceived quality but solely detects a degradation. Thus, quality is mapped to a binary value, i.e., a low quality when a degradation is detected, and a high quality if no degradation is found.

Camera shake assessment and camera misalignment algorithms use visual and auxiliary sensor features. These algorithms show the trade-offs between accuracy in detecting degradations and runtime. Whereas auxiliary sensor-based algorithms are quick to process, but usually suffer from imprecise sensors. These existing approaches neither compensate for the imprecision nor leverage visual features to support auxiliary sensor-based algorithms.

In general, it can be concluded that despite the overwhelming success of UGV, a limited set of algorithms exist to detect degradations common in UGV.

#### 2.3.3.2 *Compression and Transmission Effects*

Instead of discussing assessment methods for single impairments such as macro blocking, comprehensive objective quality metrics are discussed [**Wang2002**]. These algorithms analyze the most common and most deteriorating degradations occurring while encoding

and transmitting a stream. The section describes the objective quality assessment method Peak Signal to Noise Ratio (PSNR), Structural Similarity Index (SSIM), Video Quality Metric (VQM), Video Quality Model with Variable Frame Delay (VMAF), and Video BLIINDS (V-BLIINDS).

*PSNR*

To calculate the PSNR, a pixel-by-pixel comparison of two video frames is performed that investigates the impaired frame on noise in relation to the reference frame. Even though PSNR can be quickly calculated, it neglects features such as the "[...] content, [...]" [**Winkler2005**] and "[...] viewing conditions on the actual visibility of artifacts [...]" [**Winkler2001**]. Thus, it shows a weak correlation with perceived quality, but the PSNR is often used to assess the performance of video codecs and streaming applications [**Suarez2016**, **Winkler2008**].

The PSNR [**Stathaki2011**] is expressed as

$$\mathrm{PSNR} = 10 \times \log_{10}\left(\frac{L^2 \times r \times c}{\sum_{i=0}^{r-1}\sum_{j=0}^{c-1}[R(i,j) - I(i,j)]^2}\right) \qquad [\mathrm{dB}] \tag{9}$$

$L$ determines the maximum pixel values, which is usually 255 in a 8 bit, monochrome representation of a video frame. $i$ and $j$ represent pixel indices, $r$ represents the number of rows and $c$ the number of columns in a frame. $R$ is the reference frame whereas $I$ represents the impaired video frame.

*SSIM*

The Structural Similarity Index (SSIM) is an objective quality assessment metric that has been designed to determine the closeness of two still images [**Wang2004**]. It relies on the investigation of the luminance, the contrast and the structure in the images and achieves a good correlation with subjective study experiments. For analyzing video sequences, it neglects the impact of motion.

The SSIM is calculated as

$$\mathrm{SSIM}(R,I) = \left[l(R,I)^{\alpha} \times c(R,I)^{\beta} \times s(R,I)^{\gamma}\right] \tag{10}$$

where $\alpha + \beta + \gamma = 1$. $\alpha$, $\beta$, and $\gamma$ build weights for the individual components of the SSIM.

The three components analyze the luminance ($l(R,I)$), the contrast ($c(R,I)$) and the structures ($s(R,I)$) of two video frames $R$ and $I$. $R$ and $I$ are represented as array of pixel values of a gray image, where each pixel has a bit depth of 8 bit and a single channel.

The luminance component of the formula is calculated as

$$l(R,I) = \frac{2 \times \overline{R} \times \overline{I} + c_1}{\overline{R}^2 + \overline{I}^2 + c_1} \tag{11}$$

$\overline{R}$ is the average of frame $R$ and $\overline{I}$ is the average of frame $I$. $c_1 = (k_1 \times L)^2$ is a stabilizing factor in cases when the averages of $R$ and $I$ are close to zero. The authors state that $k_1$ should be a small constant $<< 1$ and $L = 255$ for 8 bit gray images.

The contrast component is calculated as

$$c(R,I) = \frac{2 \times \sigma_R \times \sigma_I + c_2}{\sigma_R^2 + \sigma_I^2 + c_2} \tag{12}$$

Here, $\sigma_R^2$ and $\sigma_I^2$ are the variances of $R$ and $I$. Again, a stabilizing factor is used and calculated as $c_2 = (k_2 \times L)^2$, where $k_2 << 1$.

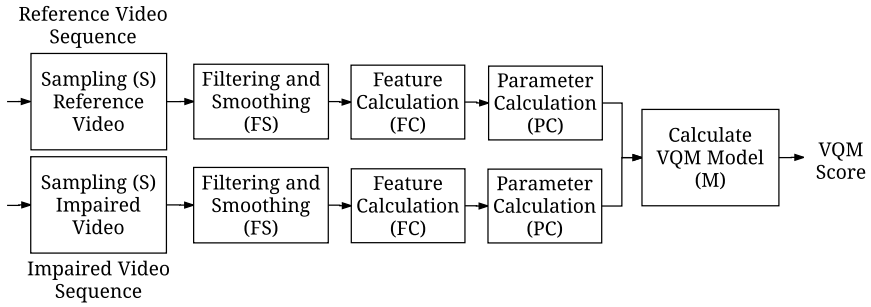



Figure 7: Essential steps of a VQM assessment inspired by Pinson et al. [**Pinson2004**].

For the calculation of the structural component, the luminance component is subtracted from the frames. $s(R, I)$ is then determined as

$$s(R, I) = \frac{\sigma_{R,I} + c_3}{\sigma_R \times \sigma_I + c_3} \tag{13}$$

$\sigma_{R,I}$ represents the covariance of $R$ and $I$. The reference implementation leverages $k_1 = 0.01$ and $k_2 = 0.03$ by default. The weight factor $c_3$ is calculated as $\frac{c_2}{2}$.

The result is a single integer in the range of [-1,1], where a value of 1 is reached when $R$ and $I$ are identical frames.

*VQM*

Pinson and Wolf propose a perceptual FR, as well as an RR objective quality metric, called the Video Quality Metric (VQM) [**Pinson2004**]. It provides an analysis of video degradations in the spatial as well as the temporal domain. A Spatio-Temporal Region (ST-Region) is extracted from a video sequence, which represents $n \times m$ pixel blocks tracked over multiple video frames. As an FR method, a potentially impaired video sequence is compared with a reference sequence. The steps of VQM are depicted in Figure 7, consisting of a Sampling (S), a Filtering and Smoothing (FS), a Quality Feature Calculation (FC), and a Quality Parameter Calculation (PC) step. The two results of the reference and the impaired version are then combined in the Model Calculation (M) step.

Reference and impaired video sequence must have the same frame rate and resolution before they are decoded into the YCbCr color space. This color space contains one luminance channel (Y) and two chrominance channels (CbCr). The first step is Sampling (S), and it is conducted for each of the video sequences independently. S converts each of the video frames into floating-point representations as it offers the required accuracy for a precise estimation of the perceived quality.

Filtering and Smoothing (FS) applies the Sobel edge filter to the synchronized frames of the reference and the impaired video [**Sobel1968**]. To analyze the quality degrading changes, an edge difference metric is calculated for a spatial metric, and a temporal metric is computed for detecting degradations.

The Quality Feature Calculation (FC) step determines visual features in a single video frame and across frames, which are combined in an Spatio-Temporal Region (ST-Region). It pans across $8 \times 8$ pixels and 6 frames at a frame rate of 30, i.e., 0.2 seconds of video.

Five different feature sets are extracted from the ST-Region, including two features that focus on structural information whereas the others address color information, contrast, and motion. The resulting ST-Regions from both representations are compared by the PC step, resulting in an ST-Region error matrix quantifying the differences between both videos.

Afterwards, the errors of each feature across the ST-Regions are condensed into a single value - and for different ST-Regions. A linear regression model determines the resulting VQM value in a Model Calculation (M) step. The regression models were validated in extensive subjective studies. The VQM values range from 0 (reference video sequence) to 1 (impaired video sequence).

VQM achieves high correlations with subjective experiments but requires a reference video for conducting the assessment. At the same time, the runtime for processing is rather high, making it unusable for real-time applications.

*VMAF*

VMAF is the recently proposed recommendation of the video provider Netflix [**Li2016**].

VMAF leverages the strengths of different metrics to predict the perceived quality by fusing them using an SVM regressor. The SVM regressor generates the weights on the impact of each metric on the final quality score. The leveraged metrics comprise the Visual Information Fidelity metric, which measures the loss of information. The second metric determines the loss of details and the loss of content visibility, which distracts viewers. Both metrics measure the spatial impact of degradations. Also, the impact of degradations on the motion is assessed by calculating the average absolute differences of the pixels on the luminance plane.

The learned model has shown to reliably detect compression and transmission artifacts for a broad range of datasets.

*V-BLIINDS*

V-BLIINDS is a NR objective quality assessment algorithm combining spatial and temporal artifact detection and achieving comparable correlation rates with subjective studies [**Saad2014**]. The algorithm leverages the concept of natural scene statistics, which relies on the observation that undistorted videos show statistical regularities. Irregularities in distorted videos allow a determination of the quality loss. V-BLIINDS leverages irregularities when representing motion in a video by using a Discrete Cosine Transform (DCT) representation of a video, or specifically of two consecutive frames. A joint spatial and temporal distortion detection and assessment is performed on the DCT representation. A two-dimensional spatial DCT is applied on $n \times n$ pixel subblocks of a video frame. Between two related subblocks, frequency differences are calculated. From these differences, a statistical analysis is performed to detect irregularities from undistorted frequency coefficients.

*Discussion*

Table 1 shows the performance of the algorithms regarding processing time for a 15 second video and the correlation with subjective assessments. This correlation gives the capability of an objective quality metric to predict the perceived quality by an average human. The performance is measured on a commodity server[9]. The basis for our results is the LIVE video dataset, which contains ten 720p videos, each available in 15 distorted versions [**Vu2011**]. Each video has been manually annotated by a quality value by 38 subjects in controlled experiments.

9 Hardware setup: Intel Xeon CPU E5-1650 with 64 GB of dedicated memory

Table 1: Comparison of objective video quality assessment algorithms. The columns depict the metric classification, the algorithm execution time in seconds for analyzing 15 seconds of video at a resolution of 720p and 30 FPS (lower is better), the SROCC with subjective studies (CC, higher is better), and the metric's recommendation status.

| **Algorithm** | **Type** | **Runtime** [s] | **Correlation LIVE DS [Vu2011]** | **Recommendation** |
|---|---|---|---|---|
| PSNR | FR | 1.6 | 0.4035 | - |
| SSIM [**Wang2003**] | FR | 104 | 0.658 | - |
| VQM [**Pinson2004**] | FR / RR | 88 | 0.770 | ANSI/ITU [**ANSI2003**, **ITU2004**] |
| VMAF [**Li2016**] | FR | 198 | 0.872 | Netflix Inc. |
| V-BLIINDS [**Saad2014**] | NR | 14.2 | 0.759 | - |

An advantage of the simple PSNR is the low execution time of 1.6 seconds. Due to the low correlation, the PSNR is only the baseline for other algorithms. None of the other FR algorithms can perform a reliable estimation of the perceived quality in real-time, which is essential in live streaming. An appropriate processing time for a live scenario is less than 15 seconds. Especially, highly precise algorithms such as VMAF and VQM have execution times far beyond real-time capabilities. VQM was validated in large-scale user studies and standardized by the American National Standard Institute (ANSI)/ITU. A recent modification of the VQM algorithm allows to run it on a Graphics Processing Unit (GPU), which achieves the same correlation in real-time. This version, called Real-Time Video Quality Assessment (RT-VQM), has been co-developed by the author of this work, but is not discussed in this thesis [**Wichtlhuber2016**].

The VMAF, as proposed by Netflix, relies on an SVM for quality estimation. Once trained, the classification should be quickly achieved. Still, the processing times are even higher than the ones for VQM. The discussed V-BLIINDS algorithm achieves a considerable correlation at a low runtime. The algorithm is thus suited for efficient UGV assessment when no reference is present.

### 2.3.4 *Summary*

Findings can be gained from the previous discussion of the recording quality and the video quality. Existing recording quality assessment algorithms detect degradations, but do not assess their impact on human perception. Quality models are needed to fill the gap to not only detect a degradation, but also to allow one to assess its impact on the perceived quality. Algorithms need to be redesigned to not only detect a degradation but also quantify the individual characteristics of each degradation.

In the live UGV streaming scenario pursued in this thesis, a real-time assessment of the perceived quality is required. The presented recording quality assessment and video quality assessment metrics are too slow for real-time processing. Exceptions are algorithms that leverage auxiliary sensors to detect recording degradations. They usually suffer from a reduced correlation with subjective studies and have to cope with noise in the sensor readings. Approaches need to be found to improve the speed of the algorithms while keeping a high correlation with subjective studies.

Existing algorithms discuss the assessment of a single video stream. None of the algorithms addresses how to scale the quality assessment to multiple video streams, as, e.g., needed in the video composition proposed in the thesis.

## 2.4 MOBILE VIDEO UPLOAD

An essential step in the video streaming process (see Section 2.3.1 and Figure 2) is the live upload of a video stream from a smart mobile device. The mechanisms that allow live upload to nearby or remote receivers are combined in a so-called Mobile Video Broadcasting Service (MBS). In the remaining section, the MBS is defined, and challenges in the scenario of mobile live video uploading are discussed. Finally, the existing MBS approaches from both academia and industry are compared.

### 2.4.1 *Description of an MBS*

The description of MBS is derived from the Personal Broadcasting Service (PBS) as defined by the 3rd Generation Partnership Project (3GPP) [**3GPP**]. The term MBS is used in contrast to PBS to emphasize the focus on mobile devices that provide live video. A PBS allows providing any media from any device.

An MBS leverages smart mobile devices capable of recording and uploading video, i.e., broadcast or multicast them to a large set of receiving users. In the case of live broadcast, the video needs to be delivered to multiple users simultaneously. The technical implementation of a broadcast is not described by the 3GPP, allowing it to be implemented on any suitable layer of the ISO/OSI network stack. For the collection and the distribution of video streams, the MBS uses network technologies supporting IP.

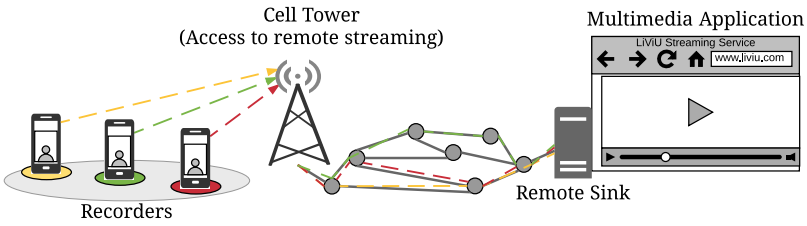
Different roles exist in an MBS. The MBS provider is an individual generating a video stream to distribute it to other users. In the remaining thesis, this role is replaced by the terms *recorder* or *recording user*. The MBS user consumes the media stream received from a recorder. A consumer can be any device, but is assumed to be another smart mobile device. In the remaining thesis, the MBS user is called a viewer. MBSs may include independent service providers, such as Facebook, which offer servers to coordinate recorders and viewers. A service provider is responsible for the media delivery between the recorders and the receivers, possibly involving intermediate servers for processing. In many cases, the involvement of such an intermediate service provider is not required. In the remaining work, service provider is represented by a server.

### 2.4.2 *Scenarios for Mobile Broadcasting Services*

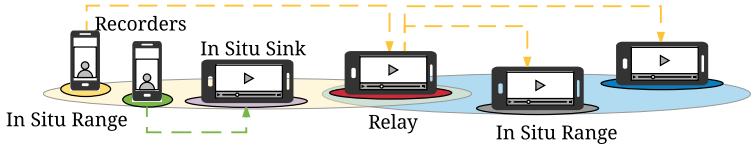
Scenarios for MBSs consist of the remote, the in situ, and the hybrid streaming (see Figure 8). All scenarios take place with no dedicated network infrastructure for the MBS; rather, they share networks with other applications and devices.

### 2.4.3 *Remote Streaming*

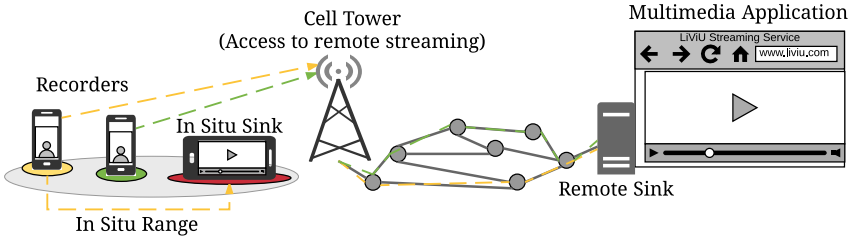
The remote streaming scenario consists of two roles: the recorder and the receiver (or server) of a video stream. Any distance can be between the recorder and the server, but an IP-based network connects both. In this scenario, the focus lies on the nearly ubiquitously available cellular networks. Their available resources are limited. LTE, for instance, achieves an average of $9.86\frac{\text{MBit}}{\text{s}}$ in 2015 in the United States of America [**OpenSignal2016**]. Universal Mobile Telecommunications System (UMTS) connections achieve in the same year and region an average throughput of $1.75\frac{\text{MBit}}{\text{s}}$ [**OpenSignal2016**]. Upload speeds of end user Internet connections are lower, as they are asynchronously configured in rela-




(a) Remote Streaming




(b) In Situ Streaming




(c) In Situ Streaming

Figure 8: Overview of the MBS scenarios discussed in this thesis.

tion to the download speed. LTE achieves upload speeds of up to $75\frac{\text{MBit}}{\text{s}}$ when the setup supports $300\frac{\text{MBit}}{\text{s}}$ on the downlink at 20 Mhz [**Ghosh2010**]. Empirical studies by Shah et al. report for 2016 that the upload speeds are very different depending on the geolocation [**Shah2016**]. Whereas in North America upload speeds of up to $23\frac{\text{MBit}}{\text{s}}$ could be achieved, the speeds in India and Pakistan ranged from 300 to $600\frac{\text{KBit}}{\text{s}}$ and in South America from 200 to $300\frac{\text{KBit}}{\text{s}}$. The latency of the connections in cellular networks are in average below 150 milliseconds, and thus not critical for the remote streaming scenario [**OpenSignal2016**].

This thesis discusses scenarios where the movement speed is limited to pedestrian walking speeds. Thus, transmissions during vehicular movements are not in the focus[10]. These speeds usually have only a slight impact on the available throughput in cellular networks. Thus, in the remote streaming scenario, mobility plays a minor role, as cellular networks shows a comparably high coverage (up to 86.73% [**OpenSignal2016**]) in developed countries, and automatic handover mechanisms exist, which guarantee a low probability of connection losses. This scenario is throughput-constrained, and rather insensitive to pedestrian mobility and latencies.

### 2.4.4 *In Situ Streaming*

An in situ scenario consists of at least one recording device and one receiving device in close proximity to each other. This distance should be less than the communication range of the wireless network technology in use. As a communication network, IEEE 802.11 is assumed to be used. Whereas in a remote streaming scenario an initial hop has to be

10 The evaluation in Chapter 7 shows that our contributions reliably work at vehicular speeds.

made to access the core network via a cellular network tower (LTE or UMTS), a regional distribution of the video is intended in the in situ streaming.

As the receivers are in close distance of each other, the focus lies on low-delay streaming. Ideally, a user receiving a video stream should not notice any delay between a live performance and the recording played back on a mobile device. In comparison to the remote streaming case, a single device may distribute the video to more than one receiver. A specific challenge in this scenario is the mobility of the devices. It does not necessarily lead to varying throughput conditions. But, if the devices move, the sender and receivers may leave the communication range. A direct communication in a single-hop communication pattern may no longer be possible. In these cases, other devices in the range of both sending and receiving devices help to establish and maintain communication.

### 2.4.5 *Hybrid Streaming*

In a hybrid scenario, mobile recorders stream media to nearby receivers as well as to a remote streaming sink, e.g., a server. Until now, no MBS has been proposed which addresses this hybrid scenario. One reason is the challenge to combine low-delay in situ streaming with a high-quality, bandwidth-constrained remote streaming.

In a hybrid streaming scenario, a single network interface may not reach all the intended receivers of a media stream. Another engineering challenge may thus be multiple network interfaces, i.e., IEEE 802.11-based and cellular networks. Appropriate MBS protocols are required for an efficient transmission of the media streams.

### 2.4.6 *Existing Work on MBSs*

#### 2.4.6.1 *Categorizing MBSs*

We now classify existing work on MBSs. The discussion focuses on assessing the abilities to cope with varying network conditions, to leverage content adaptation, and whether a prototype of the MBS was developed.

The live streaming support is essential for an MBS so that each evaluated proposal is classified regarding the live streaming support (LS). A live streaming support can be at the heart of a protocol (+), be supported (○) or not discussed (-). The scenarios (S) introduced in Section 2.4.2 needs to be supported by an MBS. Scenarios range from remote (R), in situ (I) to hybrid streaming (H).

Related to the streaming scenario is the classification of different MBS applications with respect to the supported network technologies (NW). Either IEEE 802.11 or cellular networks such as LTE or 3rd Generation Mobile Networks (3G) networks are used. Furthermore, the support for ad-hoc streaming is denoted by "A".

Mobile support (M) describes the characteristic that a proposed MBS makes assumptions which do not hold for smart mobile devices. If mobile support is not provided, no working prototype of the system can be built on retail phones (-). An existing prototype or a simulative model, which can be mapped to a prototype supports this characteristic (+).

In any of the described networking scenarios, a common phenomenon is the uncertainty about the network conditions, which includes varying delays and throughput rates. Content adaptation in the form of adaptive video streaming (AS) is a promising solution to the challenges. If a proposed MBS uses an adaptive streaming approach and realizes it on a smart mobile device, it is perceived as supporting this feature (+). Theoretical approaches

that cannot be realized with existing technology are described by a ○ in our classification. A system without support for adaptive streaming is described with a -.

How media streams are distributed can be classified into push-based (P) delivery, where the recorder sends the media stream to the viewers, or pull-based delivery in which the viewer requests the media stream (PL). The type of distribution is termed scheduling (SCH).

Besides media streams, mobile recording devices contain a broad range of sensors, which can be used to understand the context of a recording. The auxiliary data support (ADS) of each protocol is evaluated. A protocol can support auxiliary data including monitoring data, or sensor samples (+) or not (-).

The next two characteristics indicate if the proposed methods have lead to a real prototypical evaluation (Pr) of the system, and if standardized protocols (SP) are used for the design of the MBS. SP represents a list of the used transmission protocols.

Limited throughput rates are a major, but common challenge for MBSs. We evaluate the different prototypes, if they can cope with challenged network conditions such as limited upload capacities (LMU). We distinguish, if a proposed system addresses changing and poor network conditions (+) or not (-) in the protocol design.

Another challenge is the streaming delay (DS), which is critical in the in situ streaming scenario and still essential in remote streaming scenarios. It is distinguished, if the protocol does (+) or does not consider the streaming delay (-).

The proposed upload protocols are used by multimedia applications. The last characteristic describes, if the protocol supports specifications and requirements of the multimedia applications (MAR). It is distinguished if a protocol does not support (-), can in general support requirements of the application (○), or can adapt to requirement during a session (+).

#### 2.4.6.2 *Comparison of MBSs*

Table 2 gives an overview of existing MBSs and gives a comparison of the systems regarding the discussed characteristics.

*Industry Solutions*

The most widely used protocol for efficient media streaming is the Real-Time Messaging Protocol (RTMP). The protocol relies on the Transmission Control Protocol (TCP) and thus ensures in-order and error-compensated transmission of messages. It is thus a stateful media streaming protocol, which is message-oriented. RTMP establishes a reliable, low-delay end-to-end connection between a mobile device recording a digital video and a server. A streaming session is established using a three-way handshake procedure, which exchanges authentication information of both the recorder and the receiver. The rather complex handshake procedure is depicted in Figure 9. RTMP establishes a secure application layer coordination to ensure, that sender and receiver of a media stream use the same protocol version. Random bytes are sent to test the network connection and make an initial guess of the best fragment size. The fragment size determines the maximum payload of a media message. It shall ensure proper live streaming, without congesting the network or receiver. RTMP is able to transfer multiple synchronized audio and video tracks in parallel. An established connection is able to transmit media segments of variable lengths with a compressed header and thus reduces the overhead. As a consequence of the join procedure, RTMP requires a rather long time until a connection is ready for media stream

Table 2: Overview of related work for Mobile Video Broadcasting Service (MBS). Features used for comparison include LS: live streaming support; S: Scenario; NW: network access technology; M: mobility awareness; AS: adaptive streaming support; SCH: scheduling; ADS: auxiliary data support; Pr: prototype available; SP: list of standardized protocols used; LMU: limited upload capacity; DS: delay sensitive. MAR: multimedia application requirements support. +: implemented; ○: compatible; - unsupported;

| | LS | S | NW | M | AS | SCH | ADS | Pr | SP | LMU | DS | MAR |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Twitch.tv **[Zhang2015]**<br>YouNow **[Stohr2015]**<br>Periscope **[Siekkinen2016]**<br>Facebook.Live | + | R | 802.11,3G+ | - | - | P | ○ | + | RTMP | - | ○ | ○ |
| NEWSMAN **[Shah2016]** | - | R | (802.11,3G) | + | ○ | P | - | - | - | + | - | - |
| DMUS **[Zhang2008]** | - | R | - | - | - | P | - | - | - | + | - | - |
| DASH-POST **[Seo2012]** | + | R | 802.11 | + | ○ | P | - | + | HTTP | - | + | - |
| ASMA**[MinQin2010]** | + | I | 802.11,A | + | + | P | - | - | - | + | - | - |
| CoStream **[Dezfuli2012, Dezfuli2013]** | + | I | 3G | + | - | P | - | + | RTP | - | + | - |
| **[Siekkinen2016]** | + | R | - | - | ○ | P | - | - | HTTP | + | - | + |
| MoviSode **[Seshadri2015]** | - | R | 3G,802.11 | + | - | P | + | + | - | - | - | + |
| MediaQ **[Kim2014]** | ○ | R | - | + | - | P | + | + | - | - | - | - |
| SODiCS **[Ito2014]** | + | R | 802.11 | + | - | PL | + | + | - | - | - | - |
| **[ElEssaili2015]** | + | R | LTE | - | ○ | P | - | - | - | + | - | - |
| DAVII **[Johansen2009]** | + | R | - | + | + | P | + | + | HTTP | + | - | ○ |
| **[Richerzhagen2016]** | + | R,(I) | 802.11,A | - | ○ | P | + | - | - | + | - | + |

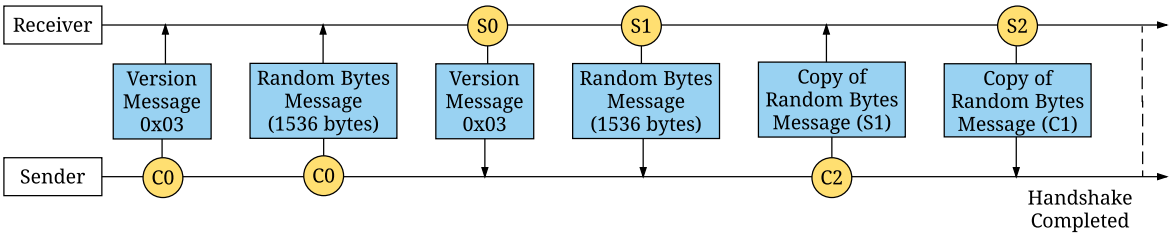



Figure 9: Joining procedure of the protocol RTMP **[RTMP2009]**.

transmissions. Today, MBSs are required which contain a join procedure that does not delay the transmission of initial video segments. An advantage of the protocol is the message structure when a connection is established. Header overhead is minimized as long as the connection exists.

Twitch.tv **[Zhang2015]**, YouNow **[Stohr2015]**, Periscope **[Siekkinen2016]**, and Facebook.Live use RTMP as a streaming protocol, and add HTTP for the transmission of auxiliary data such as network metrics or GPS coordinates. None of the approaches supports in-time adaptive streaming. The design of RTMP allows a delay to be kept low during a streaming session but the initial joining procedure is time-consuming. The system does not support any mechanisms to cope with very low upload rates. For Periscope, it is reported, that due to the streaming setup a minimum of $2\frac{\text{MBit}}{\text{s}}$ is needed for a stall-free upload **[Siekkinen2016]**.

*Mobile Video Upload*

Besides RTMP, new research prototypes have risen to compensate for weaknesses of the reliable, but rather static standard. NEWSMAN allows the upload of news videos from mobile devices **[Shah2016]**. It does not aim for live streaming in the classical sense as middleboxes are introduced, which decouple transmission to the middleboxes from the long-haul transfer to remote servers. It optimizes the scheduling of the videos created by

the recorders and selects an appropriate video quality for each recording. Quality selection and transcoding are performed on the middleboxes. A simulative study without a modeled underlying network was performed to evaluate the transcoding and scheduling of the videos. Besides NEWSMAN, a range of similar protocols exist, which do not focus on the immediate, but the delay-tolerant distribution of video streams recorded on a mobile device. One example is DMUS by Zhang et al. [**Zhang2008**].

*Support for Auxiliary Data*

MoviSode does not offer live upload of video streams, but was one of the first systems to introduce auxiliary data upload [**Seshadri2015**]. The application allows video streams to be annotated with the PoI coordinates. Remote users can query for the PoI and retrieve a list of devices offering the respective video streams. The list determines the priority of the video streams being uploaded. A similar approach is pursued by Kim et al. with MediaQ, a multimedia collection and management framework [**Kim2014**]. The proposed mobile application uses a sensor to describe when, where, and what has been recorded including the detection of PoIs. The focus of neither MediaQ nor MoviSode lies on the efficient upload in challenged networks. Thus, aspects such as adaptive streaming support or flexible scheduling are not discussed.

SODiCS can collect video in a pull-based manner from mobile devices [**Ito2014**]. The basic idea is that videos and sensor data are gathered and stored by servers or cloudlets. Mobile cameras always try to save their videos on the remote server. It copes well with any breakdown of a cellular network.

*Adaptive Live Uploading*

Media upload protocols can learn from the distribution protocols for video streams. Thus, Seo et al. discuss how the Moving Pictures Expert Group (MPEG) Dynamic Adaptive Streaming over HTTP (DASH) standard can be applied to media upload [**Seo2012**]. For this purpose, they leverage the Hypertext Transfer Protocol (HTTP) POST method to upload a segmented video stream to a server. The proposed MBS uses a server-driven adaptation and transcoding scheme and achieves a start-up delay of about the duration of one video segment in WLANs.

Siekkinen et al. extend the idea of using DASH by using an MVLC for allowing adapting video content to the upload conditions [**Siekkinen2016**]. As no prototypical real-time production of MVLC is possible, the proposed scheduling strategies are evaluated in simulations. No validated wireless channel model has been used to model the upload, but the proposed SVC-DASH shows that an optimal selection of the bit rate of each video chunk improves the continuity and the overall quality of video streams.

El Essaili et al. investigated the process of uploading a video when a central entity can coordinate the scheduling of transmissions in an LTE network [**ElEssaili2015**]. The system is described as QoE-UL for its ability to support quality-driven uploads. They show an optimal decision for the uplink transmission and determine which client should upload the video at what point in time. They optimize the resource usage and offer both low complexity as well as optimal scheduling strategies. Additionally, a centralized approach allows multi-device and cross-layer optimization. Their system is based on a significant number of requirements that cannot hold in a real deployment. For example, mobile devices are assumed to leverage MVLCs which is not possible in real-time. Additionally, the assumption that current MVLCs result in less data traffic in comparison to non-scalable encoding has shown to be incorrect for current standards [**Grafl2013**].

The DAVVI system is designed to generate video segments and upload them immediately after recording in order to generate a low-delay video streaming experience [**Johansen2009**]. Johansen et al. report how they dynamically adapt the bit rate of a video during the upload, which is achieved using a middlebox server for transcoding and segmenting. Also, DAVVI allows the distribution of metadata and uses HTTP for media upload.

*In Situ Streaming*

The idea of in situ streaming is driven by CoStream [**Dezfuli2013**, **Dezfuli2012**]. Their work describes beneficial UI design and collaboration styles when using an MBS. Their research supports the easy retrieval of media streams shared by a group in a live streaming manner. The system leverages a nearby server that mediates the streams between recorders and viewers. The upload is conducted in a push-based manner using the Real-Time Transport Protocol (RTP), achieving upload delays of around one second.

An in situ streaming scenario is used by Adaptive Strategy for Mobile Ad-hoc streaming (ASMA), which focuses on the optimal push-based scheduling of video streams encoded in different video layers using an MVLC. The method assumes that MVLC can be efficiently produced on a mobile device. This work specifically addresses varying network conditions using adaptive streaming. Specific features of IEEE 802.11 in ad-hoc mode are used to support the probabilistic scheduling and layer selection process.

A simulative model for a collaborative upload of video streams produced by phones is proposed by Richerzhagen et al. [**Richerzhagen2016**]. The system leverages multiple network interfaces to share video streams in situ and with remote receivers. The in situ distribution is used so that each device can act as a relay for accessing remote servers when no cellular connection is available. The simulation assumes that video streams are efficiently produced as an MVLC, which is not possible on smart mobile devices till today.

### 2.4.7 *Discussion*

A set of protocols exists that supports live upload of digital video. Only a few systems exist that cope with changing network conditions by integrating content adaptation (adaptive streaming support). Also, most of the proposed protocols do not address how to cope with varying application requirements and are specifically designed for a single multimedia application. The combination of content adaptation and a feature to react to application requirements promises to be beneficial for an MBS in order to cope with both limited upload capacities (LMU) but also delay-sensitive applications (DS). Most of the existing MBSs focus on a push-based scheduling and neglect the advantages of a receiver pulling video stream segments, as, e.g., proposed by SoDiCS.

Also, modern protocols for MBSs must support the transport of auxiliary data to annotate media streams with context information. Only a small set of existing protocols is capable of transmitting this data.

In summary, there is a lack of an MBS - supporting mechanisms to cope with varying network conditions - that aims for a low-delay, high-quality streaming experience and addresses the requirements of multimedia applications (auxiliary data, adaptive video streaming and scheduling).

## 2.5 VIDEO COMPOSITION

Live upload of video streams does not only offer the opportunity to apply adaptive video streaming but also another form of content adaptation: video composition. Video composition applications offer the unique opportunity to improve the perceived quality and reduce generated data traffic in challenged networks [**Arev2014**, **Saini2012**, **Shrestha2010**, **Wu2015**]. In contrast to adaptive video streaming, video composition achieves the quality increase and data traffic reduction by wisely selecting which video source should upload its video stream at a given time. Out of a set of different close-by recording devices, only one (or a subset) is actively uploading its video. Over time, different devices are allowed to upload, and the video composition algorithm ensures that exactly one composed video is created. This section discusses the background on video composition and gives an overview of related systems conducting automatic composition for UGV.

### 2.5.1 *Background on Video Composition*

Video composition is described as the "[...] arrangement of film properties, such as images [...], which create the total film [...]" [**Manchel1990**]. Many works, including this thesis, focus on the visual arrangement [**Ward2003**], and especially the video shot selection.

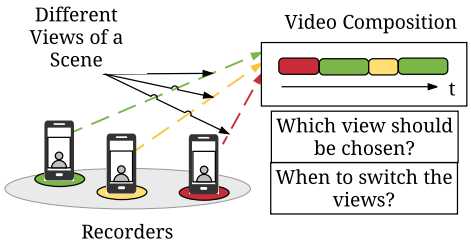



Figure 10: Video composition: Central questions when switching between different video views.

The video shot represents "[...] the smallest unit of visual information captured at one time by the camera that shows a certain action or event [...]" [**Bowen2013**]. It represents a consecutively recorded set of frames by a single camera. If the position of the camera is stable during the recording of a shot, it is classified as a static shot, whereas camera movement during the recording classifies it as motion shot [**Manchel1990**]. In a composed video, shots should be selected so that they do not distract viewers [**Bowen2013**]. Essential is the concept of "content, then form", which expresses that a composed video shall express the semantics and action intended by the director [**Manchel1990**].

Figure 10 shows the concept of video composition that leverages different video sources to create a single quality-improved video. For UGV, a wise composition selects the best view at any given time. The question as to which video view is the best at a given time is dependent on the perceived quality of individual sequences, the history of selected views and cinematographic rules. An understanding of the influences on the perceived quality in composed videos is given in Section 2.5.1.2. Also, video composition can reduce the generated data traffic in live video streaming scenarios as, ideally, only a single video source is used for video delivery.

#### 2.5.1.1 *Video Views*

It is assumed that the video composition aims at creating a video that leverages recordings capturing the same PoI. Different video sources thus capture different *views* of the same

PoI. These views differ regarding the recording position, thus, the distance and the angle to the PoI. The exact orientation of a device and the recording position can furthermore determine the shot type (often termed as shot size), which is known to have a specific impact on the perceived quality [**Bowen2013**, **Manchel1990**, **Ward2003**].

A close-up is a magnified look at an object or a person, which contains very fine-granular visual information [**Bowen2013**]. If a person is recorded, the close-up often represents the so-called "head shot", as the frame usually begins just below the chin. It is often used to depict a character's emotions [**Manchel1990**].

A medium shot reflects the common perception of a human in a close environment and is thus often used as the standard shot size [**Bowen2013**]. A person recorded in the medium shot is shown from the upper part of the legs and above. The medium shot is usually recorded at a distance of 3 to 5 meters from the PoI [**Bowen2013**].

In contrast, a medium-long shot includes the full person, possibly cutting off the feet. It allows for retrieving and identifying clothing details. At this distance, details on facial expressions and gestures are harder to see [**Bowen2013**].

For capturing the whole scene in an inclusive manner, the long shot is used [**Bowen2013**]. Between the camera and the PoI is a significant distance which allows framing the objects around a person. However, details cannot be identified.

Different approaches [**Arev2014**, **Saini2012**] have shown that the automatic, exact framing (shot size selection) is not possible. However, a good approximation is achieved when using the recording position in relation to the PoI for determining the shot size (without camera zoom).

#### 2.5.1.2 *Quality of a Composed Video*

An assumption of this work is that by leveraging different sources for constructing a composed video, the overall quality of the stream can be improved. Related composition systems support this assumption, as video composition can increase the coverage of an event and ensures diversity compared to a single video stream [**Arev2014**, **Saini2012**, **Shrestha2010**, **Wu2015**]. Also, by applying cinematographic knowledge, e.g., rules, the perceived quality of a composed video can be greater than the sum of the best parts of all single videos. Cinematography grammar supports a told story by appropriately framing the action, proliferating emotions and strengthening the storyline.

##### *Coverage and Continuity*

Single UGV streams generated by MBSs can be rather limited in duration. From UGV datasets, it is known that 90% of the recording sessions are less than 10 minutes long, where the average is 3 minutes 54 seconds [**Saini2013**]. These findings are supported by an analysis of the productive MBS YouNow. The median is slightly higher than in the aforementioned dataset with approximately 16 minutes [**Stohr2015**].

Video composition allows compensation for the fact that a single video track does not cover a full event. It does this by stitching content from different sources together into a composed video. Ideally, all single video streams overlap to some extent and still allow the composed video to cover the duration of the full event.

In contrast to the completeness, continuity describes that the temporal sequence of a real-world event shall be kept in the composed video. Furthermore, the composition ensures that each segment of a video view should have a noticeable duration, so that viewers can perceive the view change.

*Diversity of Video Recordings*

Different composition applications have shown that the major quality improvement in a composed video is the generated video view diversity [**Arev2014**, **Saini2012**, **Shrestha2010**, **Wu2015**]. Diversity in video composition is defined as the "[...] use of a variety of views in the camera selection process to increase the information content in the generated video [...]" [**Bano2015b**]. In professional productions, the diversity ensures that the perceived quality is enhanced [**Bowen2013b**, **Zettl2016**]. Shrestha et al. support this concept for UGV composition [**Shrestha2010**]. For automatic composition, some general guidelines are proposed. Saini et al. discuss that diversity should be guaranteed by inspecting the spatial and the temporal aspects of video views [**Saini2012**]. Wu et al. add that diversity should not affect motion consistency, i.e., the direction and amount of motion in a frame [**Wu2015**]. Ideally, static shots should be preferred and stitched to other static shots.

How video views should be switched is dependent on the content and genre of a composed video. In the case of music videos, rapid switching between different views leads to specific composition styles, e.g., the Music Television (MTV) composition style [**Ward2003**].

Existing composition applications report that the diversity of video views promotes the attraction of the video and avoids boredom, but it must be ensured that view jumps are not too frequent [**Arev2014**]. Professionally created video diversity is improved when cinematographic grammar rules are applied such as the "180° rule" and the "30° rule" [**Arev2014**, **Wu2015**].

*Cinematographic Rules*

Human directors learn guidelines on when a view switch shall happen and which views can be selected for composition from a cinematographic grammar [**Dmytryk1984**]. A cinematographic or film grammar describes "[...] theories that describe visual forms [...] and their functions as they appear [...] during the projection of a film [...]" [**Manchel1990**]. These rules shall be used to not distract viewers and support the storytelling of the video. Dmytryk et al. gathered rules that describe how recording devices should be positioned [**Dmytryk1984**].

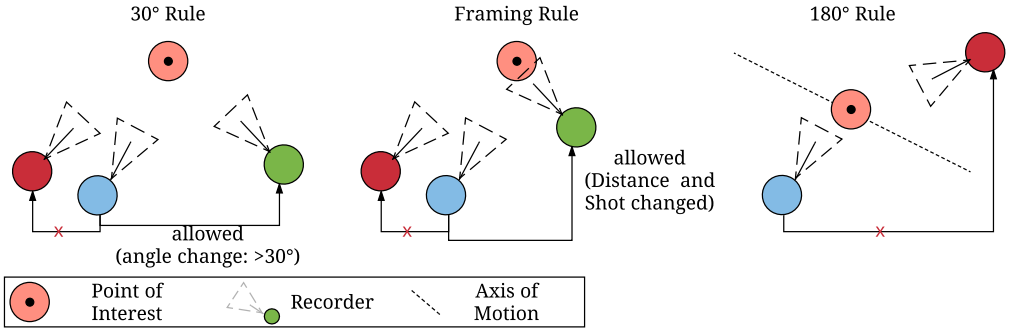



Figure 11: Illustration of the cinematographic rules discussed in this thesis: "30° rule", Framing rule and "180° rule".

Most essential is the rule: "Content, then form". It describes that the understanding of a scene being recorded helps to select a view. The story and continuity of a composed video should be more important than other cinematographic rules.

Other examples of cinematographic rules are illustrated in Figure 11. The point in time when a switch between video shots takes place is named a cut. *Jump cuts* break the continuity in a video - either spatial or temporal. Temporal jump cuts imply that between video

shots no jump in time should be perceivable for the viewer. This can be caused if the same subject is being recorded from the same or slightly varying recording positions.

Thus, when a video composition switches from one view to another, it needs to be ensured that the angle and distance of the two views are significantly different. Cuts that preserve the temporal continuity but change the recording location should comply with two cinematographic rules. The "30° rule" should be respected regarding the recording angle. It is applied if the same object or PoI is recorded in two consecutive shots. The rule recommends that between two consecutive video shots, there should be at least 30° of distance in the recording position. At stable shot sizes, this rule is especially important. The rule can be broken, i.e., the recording angle stays the same, but then the shot size needs to vary between two views [**Proferes2005**].

The second rule discusses the *framing*. From one shot to another, the framing of the two views should be different. This may lead to a long shot following a medium shot in a composed video.

The switch should not conflict with the "180° *rule*" [**Bowen2013b**]. It introduces the axis of action, which is of importance if any "[...] spatial (right-to-left or left-to-right) relationship between a character and another character or object [...]" [**Proferes2005**] is recorded. This axis connects two or more major points of a view, e.g., interacting persons. It describes the positions of objects, as well as the motion direction of objects or characters. This axis should never be crossed, as a scene should never suffer from a direction reversal when switching between views. Thus, a scene should not be captured from two opposite sides, as this may confuse the viewer. Arev et al. have shown that such a switch from one side to another can be performed efficiently within a sequence of a few switches, but not within a single switch [**Arev2014**]. For example, if objects leave a frame in a shot and return in a frame of the next shot, it needs to be ensured that they return to the same side (consistent screen direction or axis of action) [**Proferes2005**].

### 2.5.2 *Existing Work on Automatic Video Composition*

In the remaining part of this section, existing approaches towards automatic video composition are discussed, which specifically deal with UGV. The approaches are classified reaching from a manual composition to automatic composition of UGV.

#### 2.5.2.1 *Categorizing Video Composition Applications*

Our discussion of related approaches is limited to composition algorithms for UGV produced by smart mobile devices, e.g., retail mobile phones. Professional productions or static camera recording issues are not discussed. The interested reader is referred to [**Lampi2010**, **Pozzer2009**].

A video composition as understood in the thesis has to be realized for live streaming scenarios in an automatic manner. Thus, intuitive characteristics describing existing video composition approaches address their capability to automatically (*A*) compose video in real-time (*RT*).

As the video recordings are provided by an MBS, it is assessed if composition applications address that additional challenges arise such as synchronization of media streams, packet losses, and network-impaired delays. This characteristic is termed *network-awareness* (*NW*).

Table 3: Overview of related work for automatic video composition applications. Features used for comparison include – A: automatic composition; RT: real-time processing possible?; NW: network-aware; S: scalable; RQ: recording quality; VQ: video quality; AQ: audio quality; D: diverse composition; RP: recording position awareness; CR: compliance to cinematographic rules; CA: content-awareness; VS: algorithm used for view selection; CPS: algorithm used for cut point selection; F: features used for decision making. Values used are: +: implemented; ○: compatible; - unsupported; V: visual; A: audio and AS: auxiliary sensors.

| | A | RT | NW | S | RQ | VQ | AQ | D | RP | CR | CA | VS | CPS | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LACES [**Freeman2014**] | - | + | ○ | - | - | - | - | + | - | - | - | H | H | - |
| WWM [**Vihavainen2011**] | ○ | ○ | - | - | - | - | - | - | - | - | - | - | R | AS |
| MotionHMM [**Wang2008**] | + | - | - | - | - | - | - | ○ | - | - | - | ML | - | V |
| LPC [**Campanella2007**] | + | - | - | - | ○ | ○ | - | - | - | - | - | R | R | V |
| AMGS [**Shrestha2010**] | + | - | - | - | ○ | ○ | - | ○ | - | - | - | O | O | V |
| MoviMash [**Saini2012**] | + | - | - | - | ○ | ○ | - | + | ○ | - | - | ML | R, ML | V |
| TComp [**Arev2014**] | + | - | - | - | ○ | - | - | + | ○ | + | - | O | O | V |
| MoVieUp [**Wu2015**] | + | - | - | - | - | ○ | + | + | - | - | - | O | O | V,A |
| AudioCut [**Roininen2016**] | + | + | - | - | - | - | ○ | ○ | - | - | - | - | ML | A |
| ViComp [**Ban02015b**] | + | - | - | - | ○ | + | + | + | - | - | - | R | R | A,V |
| SensorComp [**Cricri2012**] | + | + | - | - | ○ | - | ○ | - | - | - | - | R | R | AS,A |

Composition applications potentially receive a rather unlimited number of video streams which need to be processed. Scalability (*S*) assesses if complex processing can be conducted in a manner which allows distribution of tasks.

Furthermore, the video stream quality can vary over time. Three additional characteristics for classifying existing systems are if the composition system inspects the *recording quality* (*RQ*), *video quality* (*VQ*) and *audio quality* (*AQ*). For these categories, a "○" represents that algorithms exist for detecting quality degradations, whereas a "+" represents approaches which quantify the perceived quality. The quantification can be achieved either by using novel quality models created in subjective studies, or by leveraging established and validated objective quality metrics.

Essential for a high-quality composition is a suitable view diversity, which implies that views can be switched during composition. The composition applications should mimic human composition, which ensures diversity over multiple switches. Thus, the composition algorithm should keep track of a history of selected views to ensure diversity (*D*) in upcoming selections. If the algorithm considers both the view selection and duration, the system fulfills a suitable diversity (+).

The diversity can be influenced by both the consideration of different shot sizes and recording positions (*RP*), complying with cinematographic rules (*CR*) as well as the system being aware of the video content (*CA*), i.e., different video genres require different composition styles.

The characteristics view selection (*VS*) and cut point selection (*CPS*) describe which underlying algorithms are used. Algorithms are classified into human decision making (H), rule-based decision making (R), an optimization problem (O), and machine-learned decision making (ML). Regarding the perceived quality of the composed video, human and machine-learned decision making (H,ML) are assumed to be the most beneficial [**Saini2012**].

Which features are used in the decision making for VS and CPS are described with the feature characteristic (F). Features are classified to be visual (V), audio (A), and auxiliary sensors (AS).

#### 2.5.2.2 *Human-supported Composition of UGV*

The manual composition system Live Authoring through Compositing and Editing of Streaming Video (LACES) shall be representative for applications, which allow the composition of UGV, but which do not offer automatic composition. LACES [**Freeman2014**] is a composition-supporting application for directors of live UGV. On a tablet, all recorded live video streams are gathered, and the director is allowed to manipulate the composed video. It is well suited for in situ streaming scenarios, as the composed video can be provided as a live video stream. Editing allows view and cut point selection, as well as frame editing and injection of non-live video.

The We want More! (WWM) system [**Freeman2014**] offers an automatic, but not very sophisticated composition. Switching between views is initiated by detecting panning using the compass of the recording devices. The view selection is not described so that a random selection is assumed. An automatic composition is possible, but the provisioning and preprocessing of the video are done manually.

#### 2.5.2.3 *Automatic Composition Application*

Wang et al. [**Wang2008**] proposed the first Hidden Markov Model (HMM)-based composition of video (MotionHMM). While MotionHMM was not designed for UGV, the model can be applied to it. It learns view switching based on detected camera motion. A cut-point analysis is omitted.

##### *Quality-aware Composition*

Campanella et al. introduced the assessment of camera shaking and motion, such as panning or tilting [**Campanella2007**]. The proposed motion assessment algorithm is successful and reliable, and it is the basis for camera shake assessment in many composition applications [**Bano2015b**, **Saini2012**, **Shrestha2010**]. The composition itself is very simplistic, solely relying on the detection of a camera shake and the video quality approximated by the brightness.

Shrestha et al. propose a quality-aware video composition application for mobile phone recordings [**Campanella2007**]. The Automated Mashup Generation System (AMGS) addresses recording quality assessment, e.g., inspecting camera shake. Video composition is seen as an optimization problem that can be solved when investigating video quality, composition diversity, and cut-point suitability. Diversity is interpreted so that each view needs to be shown at least once in the composition, even when its quality is low. At the same time, the history of views solely considers the last video shot. The optimization approach limits the application in live streaming scenarios, as it requires global knowledge of the full videos. Furthermore, Shrestha et al. [**Shrestha2007**, **Shrestha2010b**, **Shrestha2006**] made contributions towards the synchronization of different video streams using audio fingerprinting or camera flash signals, which are used in AMGS.

##### *Recording Position-aware Video Composition*

MoviMash is a composition application designed by Saini et al. being evaluated for dance and music performances [**Saini2012**]. MoviMash represents a sophisticated model as it learns compositions using a HMM, after a video and recording quality assessment. The used metrics for quality assessment are not validated with subjective studies and not reliant on established objective quality metrics. As soon as a degradation is found within a

view, the view is removed from further composition. Yet, it does not offer content awareness. A single learned model is applied to any video genre, even though it has been trained with music video compositions only. The resulting composed video shows a superior performance in comparison to quality-based algorithms, as, e.g., AMGS [**Shrestha2010**].

TComp was proposed by Arev et al. as no classical video composition, but as a video summarization application, as it integrates features for video condensing [**Arev2014**]. It is the only system that establishes precise location information on the basis of Structure from Motion (SfM) that focuses on head-mounted cameras. It uses an optimization on the basis of a trellis-graph to ensure view diversity and cut-point detection. Its computational overhead is one order of magnitude higher than required for real-time computation. The composition follows basic cinematographic rules, such as the "180° rule".

These composition applications are aware of the recording position, but no data is used to determine the quality of each recording position. Rather, the positions are used for ensuring diversity [**Saini2012**] or applying cinematographic rules [**Arev2014**].

*Investigating the Audio Track of UGV*

MoVieUp extends on MoviMash by adding an audio analysis and targeting specifically music recordings [**Wu2015**]. It introduces the "less switching principle", which shows that in contrast to video, the audio track should not be diverse. Furthermore, the audio track is assessed with respect to its quality and used for determining video view switches. The quality assessment is based on the ITU P.563 speech analysis algorithm [**ITU-P563**]. An optimization problem is formulated and solved for the video selection.

Motivated by the intensive analysis of the audio track, AudioCut focuses solely on audio [**Roininen2016**]. AudioCut is no complete video composition system, instead focusing on an audio-driven video cutting algorithm. Thus, it does not determine how to select the best view; however, it is one of the few algorithms that does not analyze the visual parts of a video. It focuses on concert recordings and shows that the cut point can be improved when inspecting the music meter and audio changes. Transitions between audio recordings can be conducted smoothly when analyzing beat difference histograms. The transitions are learned using an HMM.

The design of ViComp focuses on the inspection of the audio track as well as some visual features [**Bano2015b**]. The approach is clearly driven by audio analysis, but it also consists of a video analysis step using the subjectively validate no-reference image assessment metric Blind/Referenceless Image Spatial QUality Evaluator (BRISQUE) [**Mittal2011**]. Besides image quality, camera shaking is analyzed as a quality-degrading factor (RQ). View diversity is assumed to be sufficient if the last two video views are different from the current view. The combined factor then determines which view is selected. The cut point is determined using audio analysis, where a good cut is placed at a silent moment. Silence is detected using the spectral entropy analysis of the audio track of each video. When compared to MoviMash, this composition algorithm achieves a better quality in composition.

*Auxiliary Sensor-based Composition*

Cricri et al. introduced the first video composition application, which solely uses auxiliary sensor data to make composition decisions [**Cricri2012**]. It uses the compass to calculate on the panning and tilting of the individual cameras, and to detect the PoI by collaboratively filtering samples from different recording devices. Also, the audio track of the video is inspected. Whereas the combination allows for real-time composition of videos, it does not generate superior compositions in comparison to previously discussed work.

### 2.5.3 *Discussion*

Video composition is challenging when being realized as an automatic algorithm, as it has to consider a multitude of features and requires a significant processing overhead. Existing applications evolved from human composition to automatic quality-aware systems. Especially in the area of the quality, these algorithms neglect to consider the recording degradations or other quality effects such as the recording position. Only MoviMash analyzes the recording position in order to ensure a suitable view diversity. Their proposed approach neglects that selecting the next video view, and thus the recording position, has to comply with certain cinematographic rules. Today, location sensors can help to easily identify the recording position and estimate the shot type.

Another major disadvantage of today's composition systems is the lack of real-time suitability. The inspection of visual features of high-quality video views is time-consuming. Furthermore, the central approaches limit scalability. All algorithms assume the video processing on a central server, without any discussion of the distribution of, e.g., quality assessment tasks. Cricri et al. propose a centralized algorithm that instead focuses on auxiliary sensor data, e.g., from the accelerometer, to compose video in real-time [**Cricri2012**]. The algorithm is not yet capable of achieving a quality similar to the videos composed by approaches using visual features.

The central challenges for upcoming composition algorithms are thus to combine visual, audio and auxiliary sensor-based features.

## 2.6 CONTENT-AWARE VIDEO DELIVERY TO MOBILE DEVICES

Adaptive video streaming is able to cope with unpredictable network conditions and to ensure the delivery of the highest achievable video quality in each streaming session. This section discusses state-of-the-art mechanisms for adaptive video delivery and shows that recent proposals do not address video content inspection for increased efficiency of video delivery.

### 2.6.1 *Adaptive Video Streaming*

Adaptive video streaming allows the adjustment of the video source to the available network conditions, i.e., the throughput, during playback of a video [**DeCicco2010**]. From a conceptual point of view, different protocols have been proposed for implementing the concept, ranging from server-driven adaptation to today's predominant client-driven adaptation.

#### 2.6.1.1 *Server-driven Adaptation*

For the server-driven adaptive video protocols, the media streaming server keeps track of the state of all connected streaming clients, so it can centrally switch the video version. The approach allows fine-granular switching as the server has full control over the media and knows the upcoming video chunks, their bit rate, and the network throughput. This complexity on the server is often a bottleneck when the number of streaming clients increases. The server-driven adaptation schemes have less information on the effective throughput at the client side. As a result, the client has to regularly inform the server of the current network conditions to effectively adapt the video stream. At the same time, the processing requirements of servers increase with the number of clients, leading to the

need for server infrastructures with high processing power. Usually, server-driven adaptation implies that scheduling of a video stream is push-based. Different systems have been proposed which use server-driven adaptation, mainly on the basis of push-based streaming protocols such as RTP, Real-Time Streaming Protocol (RTSP) and Real-Time Control Protocol (RTCP) [**Fiandrotti2010**, **Liu2003**, **Wien2007**].

#### 2.6.1.2 *Client-driven Adaptation*

The state-of-the-art approach for handling video streaming systems, which are used by large-scale user groups, is client-driven [**Akhshabi2011**, **Sandvine2016**]. Each client receiving a media stream makes its own - usually independent - decisions on which video representation to stream. This decision is usually based on the network conditions measured at the client. As a result, streaming servers do not have to establish persistent connections to the clients and can avoid keeping track of the client's state. In many cases, this allows video providers to use simple and cheap web servers. Client-driven adaptation infers a pull-based scheduling.

As no data on the streaming session is evaluated centrally, each client adapts on its own, which may lead to individually optimal, but globally suboptimal adaptation results. Approaches exist that establish mediating instances to gather the knowledge and optimize the streaming across clients [**Thomas2016**].

Client-driven adaptation is the predominant adaptive video streaming concept, especially due to HTTP Adaptive Streaming (HAS) protocols. HAS represents streaming protocols which allow a client-driven adaptation during video streaming, where the underlying communication protocols are set. As an application layer protocol, it leverages HTTP; and thus, TCP is used on the transport layer. It ensures reliable, in-order transfer of video stream chunks. Artifacts in the video due to lost video chunks cannot occur. TCP's slow retransmission behavior and congestion control have shown drawbacks in comparison to User Datagram Protocol (UDP)-based streaming systems. Even though the congestion control of TCP is not ideal for video streaming, many downsides can be compensated when using content adaptation. Thus, the less efficient TCP-based streaming has been widely adopted in the industry [**Akhshabi2011**]. HAS protocols are predominant in today's IP-based video streaming [**Akhshabi2011**]. Until June 2016, 35.2% of the North American Internet traffic is caused by videos delivered using HAS [**Sandvine2016**]. It can be said that a majority of today's video streams are delivered by client-driven adaptive streaming systems, especially HAS.

### 2.6.2 *Dynamic Adaptive Streaming over HTTP (DASH)*

DASH [**Stockhammer2011**] is the most recent development of HAS. DASH is the standardized evolution of proprietary HAS solutions such as Microsoft Smooth Streaming [**SmoothStreaming2016**], Adobe's HTTP Dynamic Streaming[11] and, it is related to Apple's HTTP Live Streaming (HLS) [**hlsDraft**]. DASH standardizes the protocols for the transport of the video stream similar to HAS. It defines the description of video versions in a manifest - the Media Presentation Description (MPD). The different versions of a video that are used for adaptation are called *representations*. The standard is agnostic to how these representations are en- or decoded, but they should have different target bit rates representing different quality levels. Resulting bit rates are affected by the video resolution, the signal-to-noise level (or quantization), and the frame rate of the video. The video characteristics, e.g., the

11 http://www.adobe.com/de/products/hds-dynamic-streaming.html; Visited on: 10/06/2016

structural complexity and the motion, affect the resulting bit rate of a video. The representations of a video are split to equal duration segments and stored independent of each other.

A HTTP server is used for distributing video segments, as clients pull the segments using HTTP. The manifest eases the selection of a segment and quality as it describes each segment with the representation's resolution, bit rate, and frame rate. Depending on the available network resources, the client decides at runtime which quality to request by selecting the next segment accordingly. How to come up with an adaptation decision is not specified in the standard, but intensively discussed in research.

### 2.6.3 *Quality in DASH*

For determining the quality of individual DASH representations, it is commonly agreed that regarding quality, the highest bit rate video representation is preferred over lower bit rate representations. Thus, quality models depict the quality of a representation in relation to the highest bit rate representation [**Hossfeld2014**]. Simple approaches see a linear relationship between the perceived quality and the bit rate. Zinner et al. studied the relationship between the video clip quality and the bit rate, which approximately follows a logarithmic function [**Zinner2010**]. Besides subjective evaluations, objective FR quality metrics are used to determine the perceived quality of each video representation in relation to the highest bit rate representation. Objective quality models can be generated by objective quality assessment metrics, which are discussed in Section 2.3.3.2.

The TCP-based delivery in DASH ensures that only a limited set of degradations can occur in a video streaming session. Severe impact on the perceived quality was shown for video playback freezes (i.e., video stalling), initial playback delay, and effects of adaptations between different DASH representations.

#### 2.6.3.1 *Initial Startup Delay and Video Stalling*

Video streaming clients require a playback buffer, which stores received video segments before playback. The buffer is used to compensate throughput changes, which can lead to video stalling. Stalling occurs when the video playback buffer of a client depletes as the network throughput rate falls below the bit rate of the current video representation for a longer duration. When a streaming session starts, the client has to fill the buffer to begin playback. This initial duration is called the *initial startup delay*. It is affected by the network throughput, the streamed representation, and the buffer size. The impact of this degradation has been discussed extensively in current research. It is commonly agreed that for HAS, stalling is the most severe quality impairment, and the initial startup delay is of lesser importance [**Hossfeld2011**, **Mok2011**, **Seufert2015**].

The work of Pastrana-Vidall [**PastranaVidal2004**] is one of the first contributions showing that stalling events - especially the frequency and duration of stalling - degrade the perceived video quality. Their finding is that a single stalling with a long duration is preferred in comparison to multiple stallings of shorter duration. Additionally, video viewers prefer a periodically occurring stalling pattern with stable intervals, in contrast to unpredictable stalling patterns. This finding is supported by Moorthy et al. [**Moorthy2012**].

For HTTP-based streaming Mok et al. discusses that the stalling rate is the main cause for a reduction in quality [**Mok2011**]. Their model:

$$MOS = 4.23 - 0.0672 * L_{ti} - 0.742 * L_{fr} - 0.106 * L_{tr} \tag{14}$$

where $L_{ti}$ (L1: 0-1 seconds; L2: 1-5 seconds; L3: more than 5 seconds) is the level of the initial starting delay, $L_{fr}$ is the level of frequency of stalling events (L1: 0-0.02; L2: 0.02 - 0.15; L3: more than 0.15) and $L_{tr}$ is the level of duration of the stalling time (L1: 0-5 seconds; L2: 5 - 10 seconds; L3: more than 10 seconds). They normalize the different metrics to levels in a range from 0 to 3.

A similar model is proposed by Hossfeld et al. [**Hossfeld2013**]. It differs in terms of the coefficients but has in common with Mok et al.'s or Van Kester et al.'s model [**VanKester2011**], that the frequency of stallings has the highest impact on the perceived quality followed by the duration. The models show a high correlation with subjective studies in HAS protocols. For mobile devices such as tablets, stalling is identified as the major degradation by Floris and Atzori et al. [**Atzori2014**, **Floris2012**]. Moorthy et al. [**Moorthy2012**] extend their work, indicating that the common assumption that video stalling should be avoided at any cost is not always true. Video adaptations to low bit rates can have a more severe impact on the perceived quality in comparison to a single stalling event.

#### 2.6.3.2 *Video Adaptation*

A central task of adaptation schemes in adaptive video streaming systems is to plan representation switches in a way to not distract the viewers, or decrease their viewing experience [**Papadimitriou2007**].

##### *Influence of Video Dimension*

We define that current video encoding standards can be adapted in the spatial (resolution), the temporal (frame rate), and the quality Signal-to-Noise Ratio (SNR) - or quantization - dimension. Videos can be encoded so that adaptation is possible in each of the dimensions. In an extensive literature review, Seufert et al. show that video dimensions should be considered in an adaptation process [**Seufert2015**].

Zinner et al. [**Zinner2010**] studied the effects of adaptations in the temporal and spatial dimension of a video using objective video quality metrics. The results show that higher resolutions should be favored rather than an increased frame rate. Adaptations investigated include switches in the video resolution and the frame rate. The temporal dimension has a significant impact on the video quality when the motion in a video is high [**Ghinea1998**]. Also, the SNR dimension value can be estimated by using the bit rate as an indicator of the same content, resolution, and frame rate [**Garcia2010**, **Zhai2008**].

Resolution or spatial adaptation is the key dimension for small screens, and the impact of an adaptation is related to the respective shot type [**Knoche2007**, **Knoche2005**].

Toni et al. discuss how to perform encoding with an optimized set of video representations [**Toni2015**]. The results of the study show that up-sampled lower resolution videos can provide the same perceived quality in comparison with higher resolution videos; however, only certain video genres benefit from this finding.

In subjective studies by Zhai et al., it is shown that those statements cannot be generalized, as it is very dependent on the content of a video [**Zhai2008**]. High-motion videos prefer a temporal adaptation, whereas others prefer adaptations in the other dimensions.

##### *Impact of the Adaptation*

Zink et al. discuss that the adaptation process influences the perceived quality [**Zink2003**]. The core findings of Zink et al.'s work include that the frequency and amplitude of an adaptation have an effect on the perceived quality. The amplitude defines the number

of representations between the currently played back representation and the target one. Frequent adaptations can result in a reduced overall quality in comparison to the playback of the lowest available video representation. The amplitude of an adaptation should be kept as small as possible to avoid quality-degrading effects.

Garcia et al. analyze the effects of adaptation strategies on the perceived quality [**Garcia2014**]. Studies of different types of quality switching such as encoding, spatial, temporal, and audio switches are compared. They show that multiple gradual quality switches are preferred in comparison to abrupt variations. Frequent switches are an impedance to a good user experience, unless they allow watching the highest quality for a certain time. Nonetheless, a consistent quality level is generally preferred to variable quality.

In most situations, the best adaptation depends on the encoded video content [**Knoche2005**, **Lee2011**, **Rajendran2002**, **Wang2003**, **Zhai2008**]. When different video dimensions are analyzed for a content-aware adaptation, it leads to a higher perceived quality than using a single quality dimension [**VanDenEnde2007**].

Moorthy et al. highlight that if quality levels exhibit a small degree of separation the adaptation cannot be perceived by users; thus, adaptations can be performed in a seamless manner [**Moorthy2012**]. A similar but generalized finding is made by Ni et al. [**Ni2011**]. Viewers accept quality switches of up to four quantization steps for the quality dimension, a third of the original frame rate for the temporal dimension, and only half of the original frame resolution for the spatial dimension.

Also, Moorthy et al. and Ni et al. indicate a relationship between the frequency and the amplitude of adaptations [**Moorthy2012**, **Ni2011**]. For example, low-frequency adaptations can reduce the perceived impact if strong quality variations occur. More frequent adaptations are allowed if this allows a viewer to watch the higher video quality layer for at least one-third of the overall video duration. Both research groups indicate that a constant perceived quality is preferred in comparison to highly varying, perceivable changes.

Recently, studies show that viewers get used to adaptations in a video. A study from 2016 shows that users of a HAS stream are no longer impaired by the number of video representation switches [**Nam2016**]. In contrast, it is more important that the adaptation is conducted in a covert manner. Their finding is that the high amplitude adaptations should be avoided.

### 2.6.4 *Existing Adaptive Streaming Systems*

In the existing literature and practice, it is shown that the distribution of digital video over the Internet is driven by the demand of high bit rate content, delivered via HAS-based systems, which encounter the challenge that performing adaptations may have an impact on the perceived quality. The content of a video has repeatedly been reported to have a significant impact on the perception of adaptations and the quality of video representations [**Garcia2010**, **Ghinea1998**, **Knoche2007**, **Knoche2005**, **Zhai2008**], but no generalized rules have been reported.

Another challenge comes with the rise of mobile video streaming, leading to a clash with these requirements, as access contracts limit the availability of high-speed Internet in particular on mobile devices. The leading mobile telecommunications provider in Germany recently announced that unlimited data traffic for LTE access costs approximately 159 Euro per month[12].

12 https://www.t-mobile.de/tarifoptionen/datenoptionen; Visited on: 06/09/2016.

#### 2.6.4.1 *Categorizing Adaptive Systems*

This thesis offers a discussion of the state-of-the-art adaptive video streaming systems following certain assessment characteristics, presented here.

*Mobile device (MD)* support represents if the system design of an application considers the limitations of today's mobile devices such as reduced processing capabilities, lack of codec support, and energy limitations. Current streaming systems should cope with varying *network conditions (NW)*, as they affect the streaming experience. This can be achieved by *adapting* between different bit rate representations of the same video. The adaptation of the video is not limited to solely the bit rate but can adapt in the different video dimensions including temporal, spatial and SNR dimension.

As the predominant approach for the delivery of video, it shall be investigated whether the proposed systems can be mapped to or use HAS for media delivery - and if they infer additions or modifications of the principles of HAS, such as client-driven adaptation and the usage of HTTP.

Adaptive streaming systems can leverage the advantages of novel video *encodings*, which can be classified into SVLC and MVLC. As a result, the system can leverage essential properties of the respective encoding or be agnostic to it.

Users demand quality-aware streaming, which offers the advantage of delivering the desired quality with minimal data traffic. *Quality Metric* describes if and which objective quality metric is used for estimating the video quality.

The computational processing needs, which are implied by a quality-aware video streaming, can be enormous. The usage of objective quality metrics, the application of MVLC and the adaptation considering different video dimensions may lead to scenarios in which significant preprocessing of the video content is required. Under these circumstances, it has to be assessed if the related systems support *live streaming* with currently available technology.

Finally, the two main performance criteria for mobile live streaming shall be assessed: 1) Do the approaches focus on *quality-aware* streaming? 2) Do the applications consider the reduction of data traffic for mobile streaming clients?

#### 2.6.4.2 *Discussion of Related Approaches*

Literature and practice propose a set of adaptive video streaming systems, which improve video streaming in fixed and mobile networks. Systems aiming at quality-aware streaming are discussed in Table 4. Simple streaming systems leverage information about the bit rate to improve the perceived quality of a streaming session. Juluri et al. introduce Segment-Aware Rate Adaptation (SARA), an approach that integrates detailed information on video segments in the MPD to predict the time required for the next segment to be streamed [**Juluri2015**]. The influence of the video on the perceived quality is approximated by the relationship between the bit rates of the played back video at its highest bit rate representation. The preprocessing step adds an additional delay, which can limit the scalability of the approach in live streaming scenarios.

These approaches do not investigate the influence of streaming over mobile networks to mobile devices, as, e.g., Adaptive Guaranteed Bit Rate (AGBR) does [**DeVleeschauwer2013**]. AGBR proposes an optimal scheduler in cellular networks, which is run on the cell towers and optimizes the utilization of the available throughput. The system achieves an optimal allocation when a minimum tolerable throughput is available, and indicates a level when higher representations do not offer additional quality gains. Quality in this context is simplified to bit rates, too, but the streaming experience is improved by avoiding frequent

Table 4: Overview of related work for content-adaptive video delivery. Features used for comparison include – MD: capable for mobile devices; NW: respects network conditions; Content Dimension: which quality dimensions of a video are respected (B: bit rate only; T: temporal; Q: quantization/ SNR; S: spatial); HAS: HTTP Adaptive Streaming; Coding: Leveraging of SVLC anf MVLC; Quality Metric: Used video quality metric; Live: Live streaming support; Focus - Quality: Quality aware streaming; Traffic: Data traffic reduction. +: implemented; ○: compatible; - unsupported.

| | | | | | | | | Focus | |
|---|---|---|---|---|---|---|---|---|---|
| | **MD** | **NW** | **Adaptation (Dimensions)** | **HAS** | **Encoding** | **Quality Metric** | **Live** | **Quality** | **Traffic** |
| SARA [**Juluri2015**] | ○ | + | B | + | SVLC | - | ○ | ○ | - |
| AGBR [**DeVleeschauwer2013**] | + | + | B | + | SVLC | - | + | ○ | - |
| QDASH [**Mok2012**] | ○ | + | B | + | SVLC | ○ | + | ○ | - |
| PANDA [**Li2014**] | ○ | + | B | + | SVLC | - | + | ○ | - |
| QFAS [**Cicalo2014**] | ○ | + | B | + | SVLC | SSIM | - | + | - |
| AMES [**Wang2013**] | + | + | B | - (push) | MVLC | - | ○ | ○ | - |
| SVC over RTP [**Fiandrotti2010**] | - | + | T/S | - (RTP) | MVLC | PSNR | + | + | - |
| CASV [**Akyol2007**] | - | + | T/Q | - (push) | MVLC | custom | - | + | - |
| Themis [**Medjiah2014**] | - | + | B | - (P2P) | MVLC | - | - | + | - |
| Transit [**Wichtlhuber2014**] | - | + | T/S | - (P2P) | MVLC | VQM | ○ | + | - |
| SVC over DASH [**Hossfeld2015a**] | ○ | + | S | + | MVLC | SSIM | - | + | - |
| QoE Proxy [**Essaili2013**] | + | + | B | + | agnostic | PSNR | - | + | - |
| QoE HAS [**Devlic2015**] | + | - | S/B | ○ | agnostic | VQM | - | + | ○ |

video adaptations. The effects of adaptations are integrated into a HAS adaptation scheme by Mok et al. [**Mok2012**]. The proposed QoE-aware DASH (QDASH) system ensures that the switches are wisely planned integrating jumps to intermediate bit rate levels between representations. In their subjective studies, it was shown that these switches are beneficial. Furthermore, QDASH assumes a network probing proxy, which helps to better determine the available network throughput. Similarly, Li et al. [**Li2014**] assume such a central component in the network for their Probe-AND-Adapt (PANDA) system, which optimizes the streaming across different clients by ensuring a consistent quality.

Quality-Fair HTTP Adaptive Streaming (QFAS) pursues the same goal and proposes a system which addresses cellular network delivery; and thus, mobile streaming clients, and extends quality awareness from purely considering bit rates to applying objective quality assessment metrics [**Cicalo2014**]. Simple and quick, but also slow, and precise objective quality metrics such as SSIM are used to determine the effect of different video representations on the perceived quality. The usage of SSIM limits the applicability of the system to non-live streaming scenarios.

AMES is an example of a range of systems that leverage MVLC in combination with HAS-incompatible protocols, as they either rely on push-based delivery or Peer-to-Peer (P2P) assisted streaming [**Fiandrotti2010**, **Medjiah2014**, **Wang2013**, **Wichtlhuber2014**]. AMES supports mobile devices by an efficient transcoding of MVLC to SVLC and can therefore adapt while transcoding. The client-driven adaptation is no longer supported in such a delivery scheme. In P2P-assisted streaming, especially the approaches of Transit [**Wichtlhuber2014**] and P2PStream [**Abboud2012**] are mentioned, which leverage a sophisticated perceptual video quality metric and MVLC for video streaming.

Hossfeld et al. combine HAS and MVLC by modeling a Mixed Integer Linear Programming (MILP) and thus an optimal quality adaptation scheme [**Hossfeld2015a**]. The leveraged quality models are supported by the perceptual quality metric SSIM and are backed by a crowdsourcing investigation on the perceived quality of the video streaming sessions. Yielding the optimal solution by having global knowledge of a streaming session of an

NP-hard optimization problem. The approach is not applicable to be used in practice. It is the boundary of what a heuristic could theoretically achieve. The proposed models focus on fairness in streaming to multiple users and the effect of stalling. Also, the number of quality switches is modeled as an impact factor on the perceived quality, which has recently shown to be controllable, as long as the quality delta is small [**Nam2016**].

The approach of Essaili et al. fulfills a majority of the proposed characteristics, as it leverages video quality metrics for supporting mobile HAS clients [**Essaili2013**]. Both leverage a proxy to offload the task of quality assessment from mobile devices and are agnostic to media encoding. Essaili et al. leverage a network monitoring proxy for rewriting HTTP requests of clients and shapes the traffic between the client and the server [**Essaili2013**]. Standard HAS clients are used and centrally coordinated by the LTE base station. They are controlled so that they adapt in an optimal manner to achieve a fair distribution of network resources.

Devlic et al. propose a delivery model for video streams by optimizing video content for a target quality [**Devlic2015**]. It is assumed that the video content affects the perceived quality of a video representation in a long-running video sequence. A video optimization scheme analyzes the video sequences offline, making the approach unsuitable for live streaming scenarios. The perceived quality is estimated by using the VQM, being highly precise but computationally intensive. An optimization addresses the video content showing data savings are possible - yet it lacks, as an offline process, the consideration of network variations.

### 2.6.5 *Discussion*

What is obvious from Table 4 and its discussion are the lack of applying quality-aware streaming that considers the video content. Video content affects the perceived quality in a manner where understanding helps mobile devices to stream video at minimal data traffic. The existing approaches focus on either ensuring that a streaming session is optimized regarding its individual perceived quality, or regarding fair share of quality across streaming receivers. These streaming systems thereby focus on network conditions alone, trying to avoid stalling effects. Moreover, they assume that the highest bit rate representation of a video is always beneficial for users. Video as a medium is not addressed at all - which offers a huge potential regarding a quality-aware adaptation, as different studies show a strong connection between the content being distributed and its potential for quality adaptation. Advanced adaptation schemes are used by P2P-assisted streaming systems, which leverage quality as a good that can be shared with other clients. The approaches investigate the content being streamed using recent objective quality metrics, addressing different characteristics of a video. Those approaches are rather non-beneficial for mobile devices in cellular networks. Currently, no content-aware HAS adaptation system addresses the needs of mobile clients in cellular networks.

## 2.7 SUMMARY AND OUTLOOK ON CONTRIBUTIONS

This chapter summarizes the fundamentals and existing work for the quality assessment, the recording, uploading, and the processing and distribution of live UGV, where a special focus lies on ensuring a quality-aware content adaption. The application scenario describes smart mobile devices capturing video to live broadcast the streams to nearby and remote devices. Two forms of content adaptation are proposed: (1) adaptive video stream-

ing, which allows a switch between video versions while keeping the same content; or (2) video composition which dynamically selects the appropriate content at a given time.

Both concepts require a reliable and in-time video quality assessment, which is realized by objective quality assessment algorithms. The survey on objective quality assessment algorithms shows that existing work lacks the investigation of degradations that occur during the process of recording a video. None of the existing algorithms are based on validated quality models. These research gaps are addressed in Chapter 3 and Chapter 4 of this thesis. The existing algorithms for quality assessment, which focus on degradations occurring during the encoding and transmission of video streams, are either slow or imprecise. In addition, concepts are missing for conducting video quality assessment at scale. The reduction of the runtime of the algorithms, as well as an increased scalability, is presented in Chapter 4 and Chapter 5.

The provisioning of adaptive video streams from smart mobile devices is an unexplored research direction, as current protocols neglect adaptability of the system and incorporation of application requirements. A novel MBS is presented in Chapter 5.

Also, the state-of-the-art for the two content adaptation types adaptive video streaming and video composition are discussed. Existing video composition algorithms neglect the assessment of a video's quality, cannot leverage knowledge available from directing, and are incapable of performing real-time composition. A quality-aware and real-time composition approach is the main contribution of Chapter 6.

Finally, quality-aware video delivery by using adaptive video streaming is investigated. It is found that the content of a video has a significant influence on its perceived quality. Existing protocols lack a content-aware adaptation of digital video streams. A solution to this gap is given in Chapter 7.

# 3 VIDEO RECORDING QUALITY

The first contribution of this thesis is the analysis of quality-degrading artifacts in UGV, which are related to a recording person's limited skills or a lack of suitable equipment such as a tripod. Assessing the impact of these degradations on human perception leads to a subtype of the perceived quality, the recording quality. In detail, the impact of the degradations of camera shake, harmful occlusions and camera misalignment on the perceived quality are assessed. Different characteristics of each degradation, e.g., duration and amplitude of a camera shake, are discussed and then quantified by their influence on the perceived quality. An in-depth understanding of the recording quality is essential for any UGV application.

We leverage the understanding of the recording quality for content adaptation decisions, in particular for a video composition application that is described in Chapter 6. Furthermore, video composition relies on the availability of in-parallel recorded videos from different devices at different positions. Our understanding of video composition requires that recordings capture the same scene as different video views. In this context, no subjectively approved quality models exist, which determine the impact of the recording position. The second contribution of this work is a quality model describing the impact of the recording position with respect to its distance and angle in relation to a PoI.

This chapter describes ideas, concepts and results presented in our peer-reviewed publications [**Wilk2013**, **Wilk2014b**, **Wilk2014**].

## 3.1 QUALITY IMPAIRMENTS

### 3.1.1 *Recording Degradations*

Major degradations occurring in UGV are camera shakes, harmful occlusions, and camera misalignments.

#### 3.1.1.1 *Camera Shakes*

Due to the lack of a stabilizing tripod, small, uncontrolled movements by the recording user can lead to undesired motions captured by the video. If these motions occur continuously with varying motion directions, they are named camera shakes. Intended motions of the camera include tilting and panning. In contrast to a camera shake, these intended forms of motion do not show repeated direction changes [**Ward2003**].

Characteristics of this degradation include the amplitude, the direction, the speed, as well as the duration of a camera shake. These characteristics are analyzed concerning their impact on the quality of a subjective video quality assessment. The definition of the characteristics is as follows:

- *Amplitude*: The amplitude determines the amount of movement into a certain direction. It is measured in the portion of a frame as a percentage of the video frame added to the top and bottom (or left and right).

- *Direction:* Distinction on whether the shake is performed along the horizontal or vertical axis.
- *Speed of shake:* A relative measure of the percentage of a video frame skipped in one second of video.
- *Duration:* The delta of playback points in time between the start and the end of a camera shake in a single video sequence.

#### 3.1.1.2 *Harmful Occlusions*

Many occlusions are not perceived as distortions as they are part of a scene. In cases when a foreground object crosses the line of sight between a recording device and a background object the user is interested in (RoI), the occlusion becomes distracting. Thus, *harmful occlusions* limit the possibility to record video content. Relevant characteristics of harmful occlusions include:

- *Size:* The percentage of the video frame that is occluded.
- *Position:* In which part of the video frame does the harmful occlusion occur? The occlusion may appear at the top or the bottom of the video frame. It may arise from the left or the right of the video frame.
- *Duration:* The delta in playback time between the points in time a harmful occlusion appears and disappears in a sequence.

#### 3.1.1.3 *Camera Misalignment*

*Misalignment of the camera* represents an event in which the camera is not focusing on the commonly agreed RoI, e.g., the stage during a concert. Misalignments start from slightly drifting away from keeping the main actions in the center of a recorded sequence.

In its worst form, the recording does not capture the RoI at all. We evaluate different misalignment types, which vary in terms of their direction, the misalignment from the RoI of a frame (”Percentage”), and their duration. The characteristics are described as:

- *Direction:* The direction describes whether the camera is misaligned to the left, right, top, or bottom of a video frame.
- *Percentage:* The amount of misalignment measured as the percentage of the total video frame no longer perceivable due to the misalignment.
- *Duration:* The delta between the start of a camera misalignment and its end in a single video sequence.

### 3.1.2 *Recording Position*

Different video views of the scene can be distinguished regarding the geographic position and the orientation of the recording device. Whereas the quality models for the assessment of the *recording quality* can be applied to any UGV, the models created for the *recording position* assume that all video views capture a common AoI. To map the position relative to a single event, each position can be described in relation to the PoI as the reference point.

This position can be acquired using location providers on today's recording devices, including GPS. In the proposed models, this position is simplified to a two-dimensional model. The PoI can, for example, be the stage in a concert hall, in which performers act. The distance is being measured in meters by transforming the GPS coordinates to a Universal Transverse Mercator (UTM) model. Distances can be classified into shot types: close-up, medium, medium-long, and long shots [**Bowen2013**]. These shot types were introduced in Section 2.5.1.1. This classification allows an easier comparison than the nominal distance in meters as the latter must be assessed in relation to the size of the PoI and the objects that are of interest.

Besides the shot types, the distance and the angle to a PoI play a major role in the perceived quality. Here, a frontal face capture is depicted by 0° whereas 90° shows a recording from the side of a stage. The angles of the orientations are measured in 10° steps from the origin with 0° to 90°. The assumption is that no sensible recording can be created beyond the 90° angle. Figure 12 gives an idea of the concepts of distance and angle. A traditional, proscenium show stage is assumed. For central staging scenarios a rectangular stage is assumed where each side is evaluated individually.

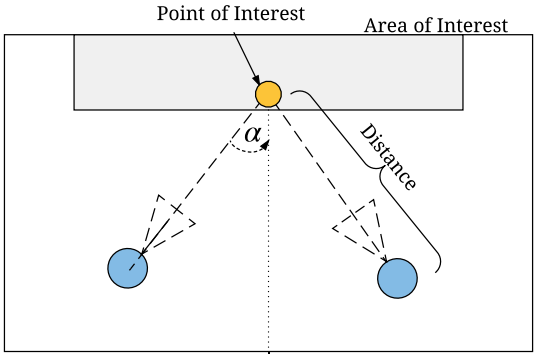



Figure 12: Illustration of measured attributes of a recording position: distance and angle.

## 3.2 APPROACH FOR CONDUCTING USER STUDIES

The quality models are generated using large-scale user studies. To achieve accurate models for the range of various degradations and their characteristics, the concept of crowdsourcing is used. Lab experiments ensure the validity of the generated models, as they are conducted in a controlled environment. A comparable UI and the same rating methodology and experiment setup were used for the studies.

The studies are conducted following the principles defined by ITU-R BT.500 [**ITU-R2012**] and ITU-T P.910 [**ITU-J2008**] for making subjective video quality assessments. All users watch multiple video sequences of a length of 8 seconds to 12 seconds. An SSCQS is used, which allows users to assess each video individually. Details on the fundamentals for subjective studies are given in Section 2.3.2.

### 3.2.1 *Crowdsourcing*

Ratings for quality models are gathered in large-scale studies using the concept of crowdsourcing. It is applied in a manner so that the quality assessments are distributed to a random set of people mediated by a crowdsourcing platform. Users rate the quality of impaired video sequences in relation to a reference. All users are compensated for their

work. The crowdsourcing mediator Microworkers[1] provides the respective crowdworkers. The evaluations leverage a web-based system, which allows to play back a stall-free video segments and offers users to rate the shown video sequences.

#### 3.2.1.1 *Recording Quality*

The recording quality assessment consists of 16 crowdsourcing runs with an average of 101 workers each. The crowdsourcing task includes to watch six video sequences in random order, including an unknown reference video. Users rate the video sequence quality on the SSCQS [**ITU-R2012**]. Thus, the task of each worker is to detect and rate degradations in the video sequences. Tasks are clustered in so-called campaigns. Each campaign represents a distinct set of video clips from one specific genre, which allows us to calculate consistent results under similar conditions. A qualification task is designed in which multiple degradations had to be found and rated. These qualification tasks train the workers and provide quick feedback on the reliability of the users. Only workers who successfully completed the qualification task are invited to participate in the evaluation. These workers are granted access to campaigns for assessing both the recording quality and the recording position.

#### 3.2.1.2 *Recording Position*

The aim of this experiment is to build accurate models on the impact of the recording position on the perceived quality. Thus, the crowdsourcing experiment asks workers to watch eight randomly ordered video sequences of the same event. The task of the workers is to judge the perceived video quality of each sequence using the SSCQS recommended by the ITU [**ITU-R2012**].

Each campaign represents a distinct set of video sequences from one of the genres: ”sports,” ”music,” ”show,” or ”scenery”. Each scene is recorded from different distances and angles that result in eight evaluated events and 79 sequences. The order of the video sequences is randomly selected for each user. In combination with a large number of workers and tests, this leads to a reduced biasing of the subjective ratings. In total, 451 workers watch and rate the video sequences, resulting in 3160 ratings.

#### 3.2.1.3 *Lab Validation*

The lab experiments ensure that crowdsourced quality models are valid even in a controlled environment, since in crowdsourcing experiments, environmental conditions and a subject’s health condition cannot be controlled. Lab experiments are conducted under our supervision and follow the recommendations of the ITU [**ITU-R2012**] regarding display size and lighting conditions. After playback of a video sequence, a five-second rating time is given. No data from the training session or qualification test is used for the final results.

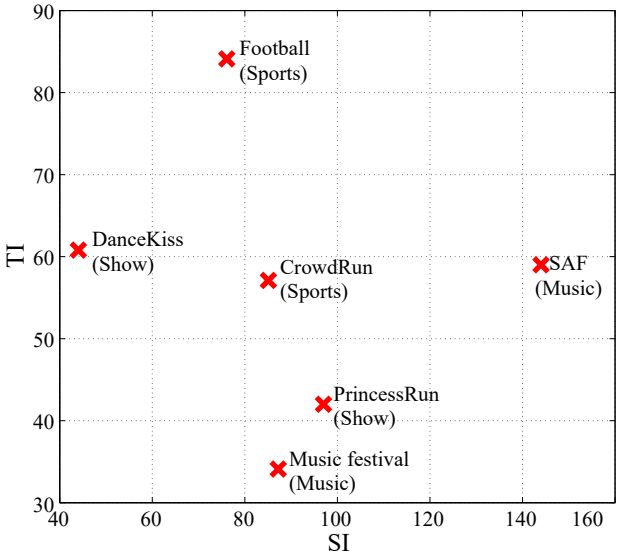
Lab experiments for the recording quality assessment consist of 16 test subjects, and 15 test subjects are recruited for assessing the recording location. A well-illuminated room with blinded windows is used in conjunction with a 42-inch display with a 720p resolution. As lab experiments are costly, only a limited set of characteristic combinations, e.g., the impact of the speed of camera shakes on the perceived quality, are evaluated.

1 www.microworkers.com; Visited on: 09/24/2016

### 3.2.2 *Evaluated Videos*

#### 3.2.2.1 *Recording Quality*

The videos included sequences from different datasets and from different genres. Here, the high-definition video dataset [**Keimel2010**] of TU München as well as the JIKU video dataset [**Saini2013**] provide different high-quality video sequences. Also, 349 of the video




(a)

Figure 13: SI and TI for the video sequences used for assessing the impact of recording degradations on the perceived quality.

sequences are recorded during a music festival and football matches in Darmstadt and Frankfurt. The JIKU dataset is a realistic set of recordings of a live event that shows degradations, such as shakes or occlusions. As the JIKU dataset does not include all degradations in the required fine granularity, videos from the TU München dataset are artificially impaired. Traces of potential shakes or occurrences of occlusions were retrieved from the JIKU dataset. The 720p resolution versions of the TU München dataset allow using lossless video information as well as comparisons between the reference and the impaired video sequences. All video sequences have a duration of 9 to 12 seconds. Videos from the TU München dataset are re-encoded in a lossless manner using H.264/AVC high 4:4:4 profile. All videos are sampled down to a resolution of 704x576 (4CIF) for the crowdsourcing experiments. In total, 1090 video clips from the three genres of sports, entertainment (no music) and music are used. The videos include different levels of structure and motions. The average Spatial Perceptual Information (SI) and the Temporal Perceptual Information (TI) [**ITU-J2008**] characteristics of all reference videos are presented in Figure 13. SI depicts the structural complexity of a video sequence represented by the edges present in the frames of the sequence. TI gives the amount of motion in a video sequence, described as the displacement of edges in consecutive video frames. Details on the calculation of SI and TI are given in Section 6.4.

#### 3.2.2.2 *Recording Position*

For the recording position assessment, new video sequences had to be created by recording different events in parallel. The videos are recorded during a motorbike race and a

soccer game (genre: ”sports”), a concert (genre: ”music”), two entertainment events (genre: ”show”), and points of interest in different German cities (genre: ”scenery”).

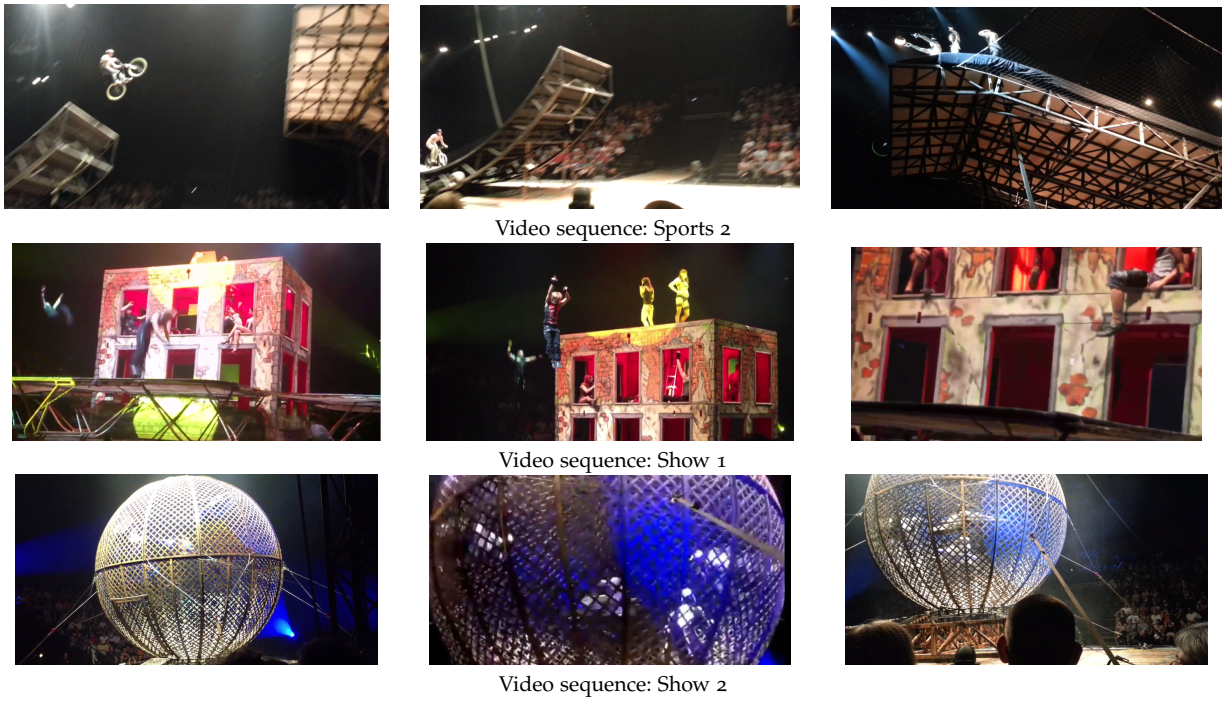



Figure 14: Impressions of the video datasets used to evaluate the perceived quality based on the recording position.

Examples of the videos are shown in Figure 14. ”Show” and ”sports” videos have been recorded during live performances in Darmstadt and Frankfurt, Germany. The videos include a circus comedy event (”Show 1”), an artistic performance, including rapid movements due to jumps (”Show 2”), and a concert with a crowded audience (”Music”). Sports events recorded include a soccer game (”Sports 1”) and a motorbike competition, including jumps (”Sports 2”).

All recordings contain a collaboratively determined RoI, which differs regarding the viewing angle, distance - and thus the perceivable level of detail. Scenery sequences are taken in Paris, France showing the Eiffel Tower in different views (”Scenery 1”), in Darmstadt showing a historic building (”Scenery 2”) and another PoI in Darmstadt (”Scenery 3”). Similar to the videos used in the recording quality assessment, all video sequences have a duration between 9 and 12 seconds. Audio tracks are removed from the videos. Video sequences are compressed at their recording and have a resolution of 4CIF similar to the other study.

## 3.3 RESULTS FOR THE DEGRADATIONS

The results of the conducted subjective surveys are quality models for camera shake, harmful occlusions and camera misalignment, as shown in Table 5. To generate the quality models, the ratings for each video sequence are gathered, normalized, and aggregated to ratings for all different characteristic combinations for all the three investigated degradations. A model for each degradation is fitted by a linear regression on the normalized ratings for classified videos. $R^2$ describes the coefficient of determination and is a metric to show the validity of the fitted model. Values close to 1 are favored. The proposed linear

Table 5: Linear quality models for different video genres impaired by camera shake, harmful occlusions, and camera misalignment.

| | **Camera Shake** $a_0 + a_{ampl} * x_1 + a_{dur} * x_2 + a_{speed} * x_3$ | | | | | **Harmful Occlusion** $b_0 + b_{size} * y_1 + b_{dur} * y_2$ | | | | **Camera Misalignment** $c_0 + c_{perc} * z_1 + c_{dur} * z_2$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Genre | $a_{ampl}$ | $a_{dur}$ | $a_{speed}$ | $R^2$ | MSE | $b_{size}$ | $b_{dur}$ | $R^2$ | MSE | $c_{perc}$ | $c_{dur}$ | $R^2$ | MSE |
| Sports | -2.572 | -0.049 | -2.412 | 0.757 | 0.09 | -5.335 | -0.109 | 0.92 | 0.057 | -2.7 | -0.09 | 0.93 | 0.012 |
| Music | -0.873 | -0.068 | -2.961 | 0.742 | 0.089 | -4.35 | -0.09 | 0.77 | 0.168 | -1.82 | -0.04 | 0.62 | 0.18 |
| Show | -0.667 | -0.074 | -3.098 | 0.827 | 0.082 | -4.963 | -0.094 | 0.91 | 0.058 | -2.83 | -0.12 | 0.73 | 0.13 |

**Note:** Camera shake: $a_{ampl}$ - Amplitude [0-1], $a_{dur}$ - Duration [0-12 seconds], $a_{speed}$ - Speed [0-1]
Harmful occlusion: $b_{size}$ - Size [0-1], $b_{dur}$ - Duration [0-12 seconds]
Camera misalignment: $c_{perc}$ - Percentage [0-1], $c_{dur}$ - Duration [0-12 seconds]

models achieve an $R^2$ of between 0.62 and 0.93, where solely the camera misalignment model for the music genre achieved an $R^2$ score of less than 0.73. $R^2$ predicts the variances in the quality assessments of our studies depending on the characteristics being assessed. Thus, it depicts how well the proposed quality models describe the perceived quality for the different video genres.

In the remaining section, these models are discussed in more detail. An example of the influence of various degradations on a single video is given in Figure 15.

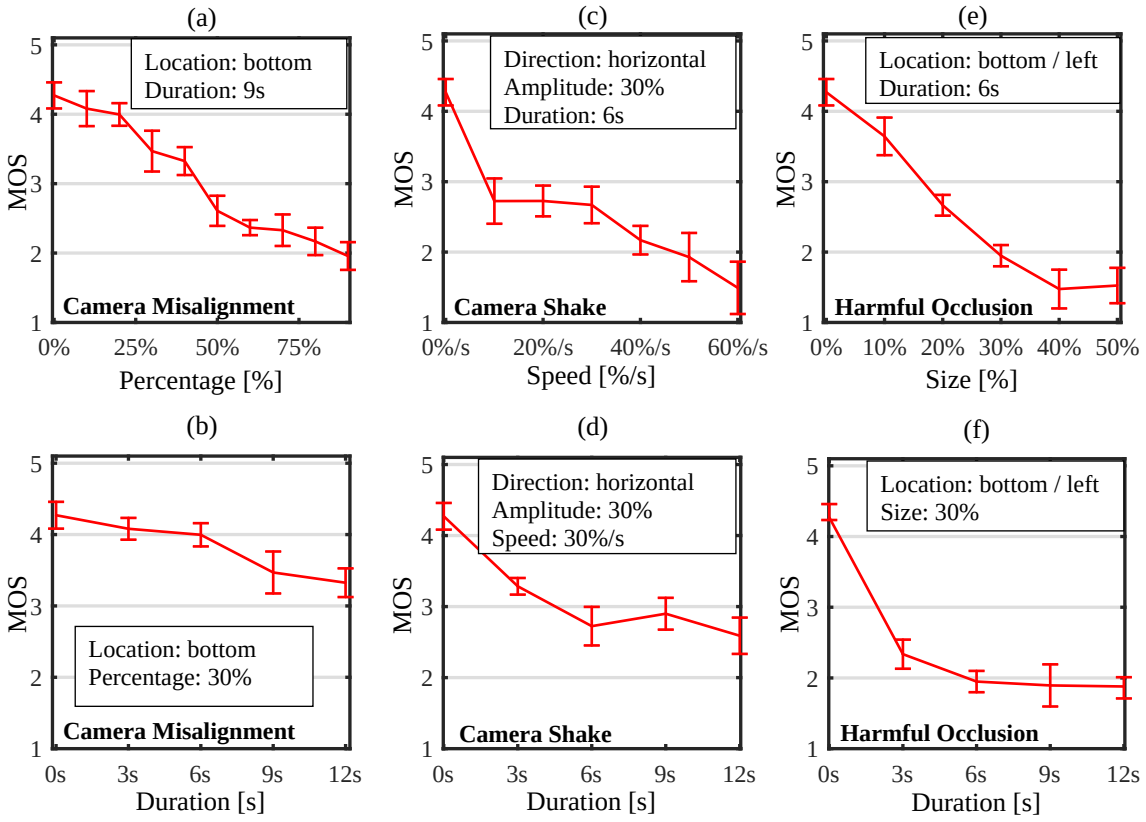



Figure 15: MOS on perceived quality reduction due to Camera Misalignment (a)-(b), Camera Shake (c)-(d) and harmful occlusion (e)-(f). The upper row shows the reduction of the MOS depending on the intensity of the degradation whereas the bottom row shows the influence of the duration of a degradation (95% confidence intervals - video: "PrincessRun").

### 3.3.1 *Camera Shakes*

Figure 15 depicts an example for the video sequence "PrincessRun", showing the relation of the speed of a shake $a_{speed}$ and its duration $a_{dur}$. In comparison with the harmful occlusions [(e)-(f)], it shows that the decrease in the perceived quality of short and slow camera shakes [(c)-(d)] is higher than the effect of a middle-sized harmful occlusion. The characteristics investigated for camera shake include $a_{ampl}$ for the amplitude of the shake, $a_{dur}$ as the duration of the shake, and $a_{speed}$ as the speed of the degradation. Results clearly indicate that all the characteristics discussed have a significant negative impact on the perceived quality. As a result, one can say that the presence of camera shake alone can reduce the perceived quality to a level not acceptable for viewers. This means that with increasing amplitude, duration, and speed of the camera shake, the perceived quality decreases. The duration has only a small impact on the quality decrease. This indicates that the pure existence of a slow camera shake over a longer time does not degrade the perceived quality in the same manner as a short but intensive shake.

Even though the different characteristics cannot be easily mapped to one another, our results indicate that fast camera shakes quickly degrade the quality even more than other degradations.

#### *Impact of the Genre*

Another observation is that a camera shake is perceived differently for different genres (see Table 5). Whereas "entertainment" and "music" sequences have similar characteristics, the amplitude of a shake in sports videos has a different effect on the perceived quality. The interpretation of this observation is that sports viewers are used to shaky recordings. The duration of a camera shake has a slightly decreased impact in comparison with other genres. The amplitude determines whether the RoI is captured in the whole sequence. A high amplitude means that during the shake the movement extends to a point where the RoI is lost. A severe decrease in perceived quality is observed in sports videos which usually focus on one distinct person, e.g., the leading ball player in soccer or a single driver in a motorbike race. The increased amplitude leads to a loss of focus on the distinct person.

#### *Direction of a Shake*

Another factor with only a small effect on the perceived quality is the distinction between horizontal shakes (uncontrolled panning) and vertical shakes (uncontrolled tilting).

Figure 16 illustrates that only in very few cases a difference between horizontal and vertical shakes for the sequence "DanceKiss" can be observed. It can be concluded that camera shake algorithms can neglect to model the direction detection. Similar findings are made for the other degradation types (see Figure 16). As a result, the quality models proposed in Table 5 neglect the direction as a characteristic.

### 3.3.2 *Harmful Occlusions*

An occlusion reduces the information a viewer can extract from a video sequence. Also, it is determined whether differences exist between different video genres. While evaluating harmful occlusions and their effect on the quality, the first attempt targeted occlusions at any position in the frame. Initial results show that spontaneously appearing objects at the border of the frame do not affect the quality at all. This holds as long as the RoI of a video

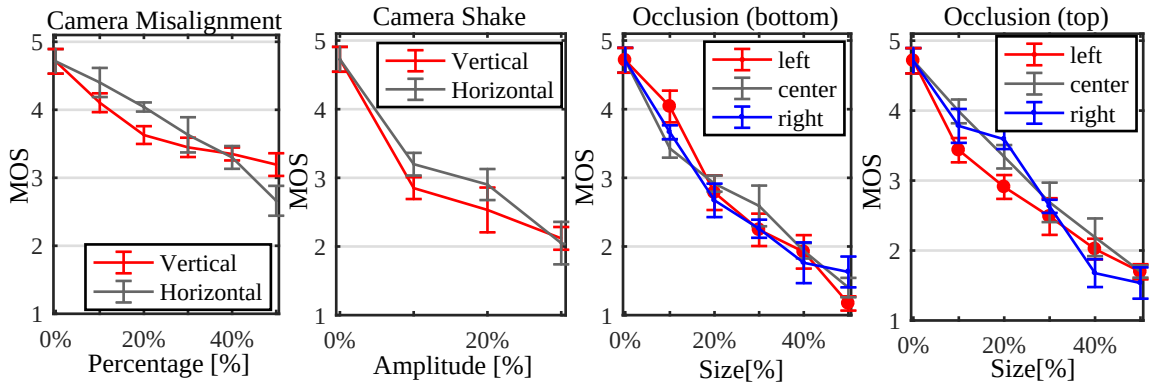



Figure 16: Influence of the location where a degradation occurs on the perceived quality for the video sequence "DanceKiss". The Figure includes the MOS and the 95% confidence intervals.

frame is not affected. Only objects positioned in the line of sight between the camera and the RoI are regarded as harmful.

Figure 15 (e)-(f) shows the reduction of quality depending on the size of the occlusion and the duration it is visible in a sequence. As mentioned, the size determines how much of the RoI is occluded. The figure illustrates quite well that short, harmful occlusions or only small sizes of the occluding object reduce the quality by a limited amount. Especially with increasing occlusion sizes, a rapid decrease in the quality can be observed. Occlusion sizes of 50% result, independent of their duration, in a major reduction of the MOS to values between 1 and 2. The observation is validated for the remaining video sequences. Table 6 shows the MOS for increasing occlusion sizes of the occlusions for different video genres. For the selected video sequences, it shows a steady decrease of the quality for increasing sizes. The position of the occlusion has a limited impact on the perceived quality. Figure 16

| Size \ Genre | Sports | Music | Show |
|---|---|---|---|
| 0% | 4.6 | 4.5 | 4.4 |
| 10% | 3.68 | 3.74 | 3.37 |
| 20% | 2.96 | 3.32 | 3.23 |
| 30% | 2.78 | 2.61 | 2.24 |
| 40% | 1.96 | 2.15 | 1.63 |
| 50% | 1.33 | 1.56 | 1.57 |

Table 6: MOS of different video genres impaired by harmful occlusions with varying size (duration: 6 seconds; position: bottom-center])

shows the difference for the sequence "DanceKiss" and for different locations at the bottom of the RoI. Similar results are obtained from the top of the RoI or any other region in the frame.

### 3.3.3 *Camera Misalignment*

The camera misalignment has only a limited impact on the perceived quality. As the reference video is shown at least once during a task, it was easier for the workers to decide when a video sequence was recorded with a misaligned camera. Still, it is remarkable that especially the small pans or tilts of 10% - 30% result in a linear but limited decrease of the MOS. In these cases, the perceived quality is still around 3.5, which indicates a still accept-

able overall quality. For the misalignment to degrade the perceived quality of a sequence to an undesirable level, the misalignment must affect the RoI of the video.

Figure 15 (a-b) shows the resulting MOS on the percentage of misalignment from the origin of the sequence "PrincessRun". The figure supports this observation. For the camera misalignment a decrease of the MOS can be observed, but especially in the range of 10% - 30%, there is no significant reduction observable. Additionally, Figure 16 shows an investigation of the difference between the vertical and the horizontal misalignment. For the "DanceKiss" sequence, the quality degradation is higher for horizontal misalignments (at 40%-50%), i.e., a panning of the camera. The reduction of MOS is observable especially in the range of 30% - 50%, and it results in a quality (MOS) of around 1.5 for video sequences with up to 100% misaligned recordings.

### 3.3.4 *Existing Quality Algorithms*

As full reference metrics are not applicable, if a video is degraded during the recording, the analysis discusses the no reference video quality metric of Yang et al. [**Yang2005**] and the recently proposed V-BLIINDS algorithm [**Saad2014**]. Both metrics show reliable quality assessment in detecting compression effects. They suffer in detecting the recording quality degradations discussed. For the most severe degradation of camera shaking, Yang et al. [**Yang2005**] found a result of an average correlation of 0.31 across the genres. Even V-BLIINDS, which outperforms established full reference algorithms, achieves a correlation with subjective assessments on average of 0.608. These findings illustrate a need for objective, NR metrics measuring the impact of recording degradations and underlines the importance of this work.

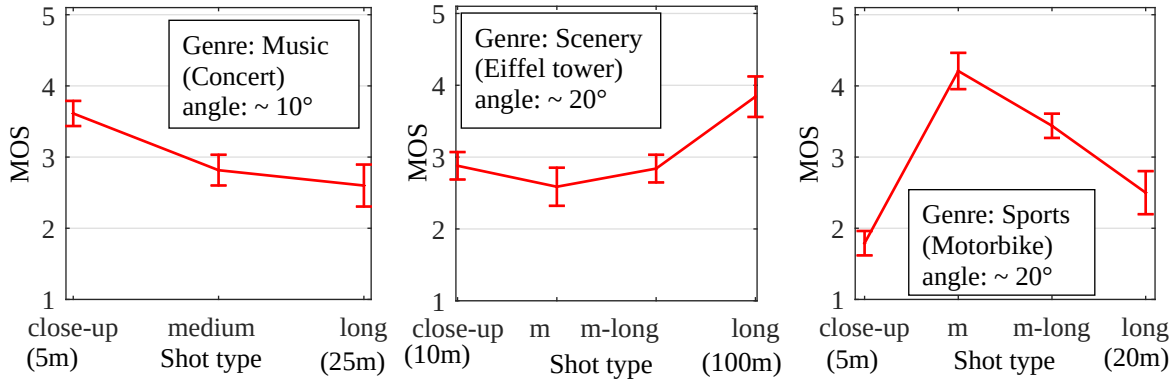
## 3.4 MODELS FOR THE RECORDING POSITION

The results describe the distance to a recorded event, classified by the shot type and the angle under which it was recorded.

### 3.4.1 *Impact of the Distance*

The impact of the distance to the AoI is important for the perceived quality. Figure 17 shows the distribution of the perceived quality for varying distances using a constant angle. It indicates that the distance has an observable effect for different recordings. For the "show" and "music" genre, results indicate that the close-ups are seen as the preferred shot type for short music recordings. A close-up represents a recording showing the main performer, e.g., in the concert recording. The quality difference between the close-up to the other shot types is small (see Figure 17 a) and nearly indistinguishable. Distances ranging from 5 up to 30 meters are preferred in all "show" and "music" sequences. Distances beyond 30 meters lead to a significant drop in the perceived quality due to the frame size of the videos and the technical limitations of smartphone camera sensors (genre: "music").

Also, the workers are asked to annotate the RoI. In most cases, it shows a larger region of the scene, but the subregion capturing the close-up is preferred in music clips. A more significant drop of the perceived quality is observed for the sequences in the genres "scenery" as well as "sports". In the case of the scenery sequences, e.g., the video sequence "Scenery 1" (see Figure 17 b), the perceived quality increases with increasing distance, as in the long shot, the complete RoI can be seen. "Scenery" recordings include a wider bor-




(a) Music - Concert (b) Scenery - Eiffel tower (c) Sports - Motorbike

Figure 17: Perceived quality of recordings from different distances with the same orientation for (a) Music, (b) Scenery and (c) Sports.

der region around the RoI in comparison to the other sequences, which indicates that the viewers are more interested in the surroundings of the PoI. For the "sports" recordings, medium shots showed the main actor, e.g., the soccer player leading the ball, or the motorbike rider performing stunts. This distance is preferred in comparison to close-ups or far distant overview shots (long shots). It indicates that a preferred shot type for the user-generated "sports" clips is recorded in a medium shot distance, allowing a combination of an overview as well as a close connection to central actions in a scene. The figure, as well

Table 7: Best recording distances for varying video genres.

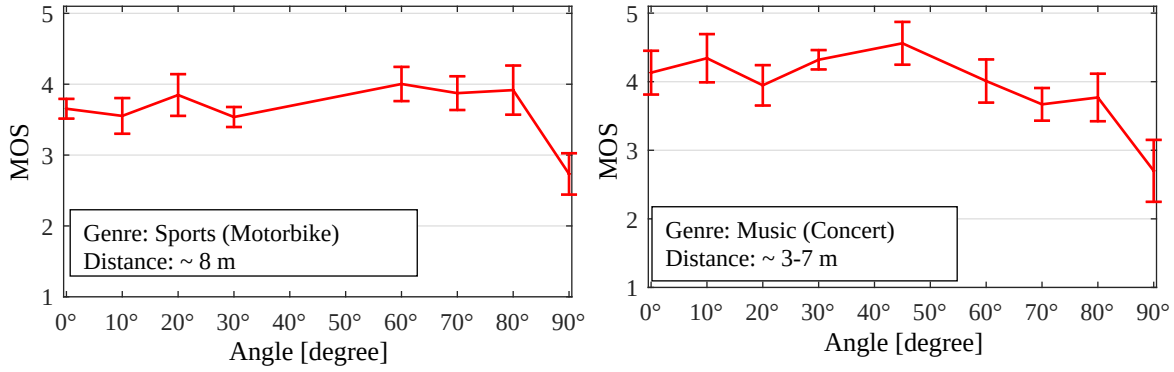
| Event | Content | Preferred shot |
|---|---|---|
| Show 1 | Circus show | medium (15 m) |
| Show 2 | Artistic show | medium (15 m) |
| Music | Concert | close (5 m) |
| Scenery 1 | Eiffel tower | long (100 m) |
| Scenery 2 | Historic building | long (60 m) |
| Scenery 3 | Statue | medium (17 m) |
| Sports 1 | Soccer | medium (15 m) |
| Sports 2 | Motorbike | medium (11 m) |

as the evaluations of the other sequences (see Table 7), let us conclude that a sweet spot for an optimal distance can be determined depending on the genre of the video.

In most of the cases only slight differences in terms of the perceived quality between recording distances can be observed. The degrading effect when selecting the *wrong* recording distance is limited.

### 3.4.2 *Impact of the Recording Angle*

The recording angle is the second characteristic being evaluated. The perceived quality represents similar quality levels in all genres for angles between 0° and 70°. Differences in the perceived quality may result from variations in the recorded videos that cannot be avoided in UGV. In extreme cases, between 80° to 90°, significant quality drops can be perceived (see Figure 18). Similar results were found for all genres, indicating that the recording angle does not significantly increase or decrease the perceived quality.




(a) Sport 2 (b) Music

Figure 18: Perceived quality from different recording angles at a similar recording distance.

## 3.5 VALIDATION WITH LAB STUDY

As mentioned, a lab experiment was set up to determine if the proposed models are also valid in controlled environments.

### 3.5.1 *Recording Quality*

We want to determine the validity of the proposed, crowdsourced models with results from controlled experiment setups. We compare the proposed quality models with results from a lab experiment for the video sequences ”CrowdRun” in the genre ”sports” and ”DanceKiss” in genre ”show”. A metric to describe the correlation between the two experiments is the Pearson correlation coefficient. The Pearson’s coefficient calculates how much values scatter around a linear trend. A function is derived in which the MOS determined in the crowdsourcing experiment describes the x-values and the lab experiment results the y-values of a linear function. The MOS of the lab and the crowdsourcing experiments must therefore follow a linear function, when compared with each other. This linear trend is depicted in Figure 19 (a,c,e). From the results gathered for the video ”CrowdRun” a linear trend can be derived. Furthermore, a high correlation is shown for all sequences, as values above 0.961 indicate no significant difference between the results of the crowdsourcing and the lab experiments.

From Figure 19 (b,d,f) it can be concluded that the crowdsourcing evaluation shows a big overlap with the conducted lab experiments (video sequence: ”CrowdRun”), and thus our results describe the relation between the degradations and video quality in a valid manner.

### 3.5.2 *Recording Position*

Also, the models derived for assessing the perceived quality in relation to the recording distance are validated with a lab experiment. The quality models from the genres ”music”, ”scenery” and ”show” are compared with the results from the lab experiments. We validate only the impact of the recording distance in both the lab and crowdsourcing experiments. Again, Pearson’s coefficient is used for describing, if a linear correlation exists between the lab and crowdsourced experiments.

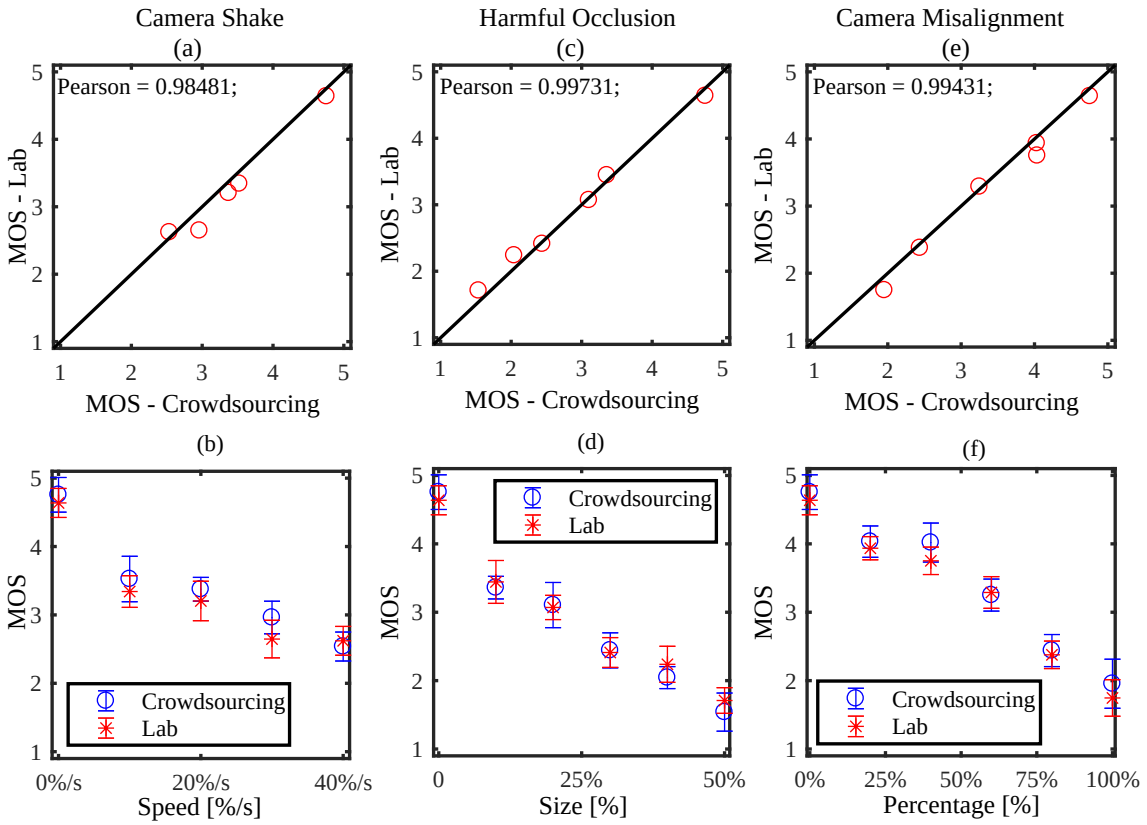



Figure 19: Correlation: Pearson coefficient and Confidence intervals (95%) of MOS for the video sequence "CrowdRun". Correlation and overlap of MOS between lab and crowdsourcing experiments for Camera shakes (a)-(b), Harmful occlusions (c)-(d) and Camera Misalignment (e)-(f)

Especially for the genres "show" and "scenery" a high Pearson correlation of above 0.9854 and 0.937 is shown. The "music" sequences show slightly differing results, especially for the reduced quality for increasing distances. As a consequence, the correlation for the genre music drops to 0.7137. In comparison to the crowdsourced tests, increasing distances are perceived to degrade the quality more. Even with a minimum correlation of 0.7137, it can be concluded that the retrieved quality models can be validated in both lab and crowdsourced experiments.

## 3.6 CONCLUSION

This chapter introduces the first quantified models on the impact of camera shaking, harmful occlusions, and camera misalignment on the perceived quality of videos. It is shown that camera shakes have the highest impact on the perceived quality; harmful occlusions reduce the quality nearly as much as camera shakes, but only if they occur in the RoI. In contrast, camera misalignments are perceived as less disturbing. Besides the impact of the degradation, individual characteristics are also assessed, such as the duration and speed of a camera shake. Depending on the video genre as well as degradation type, different characteristics have a degrading impact on the perceived quality.

Also, video composition applications can leverage different video views being recorded in parallel. These views differ regarding the device's position in relation to the recorded scene. In crowdsourced subjective studies with several hundred of workers, quality models

are created for different recording positions and video genres. The models indicate that an increasing distance to a scene degrades the perceived quality, whereas the relative angle to the scene plays only a minor role.

The proposed models are used for the creation of automatic quality assessment algorithms, which are discussed in Chapter 4, and for the video composition algorithm introduced in Chapter 6.

# 4

# SCALABLE AND ADAPTIVE VIDEO QUALITY ASSESSMENT

This chapter introduces novel algorithms to automatically analyze and quantify the impact of recording degradations on the perceived quality in a highly precise manner and with a low runtime. The classical video-based analysis is extended by auxiliary sensor data, such as accelerometer or gyroscope data, which is available during the recording of a video. Whereas video-based approaches are usually highly accurate, they require significant computational time, which makes most of them useless for real-time applications. Auxiliary sensor-based approaches offer quick results, but their performance degrades when the readings are inaccurate. This chapter introduces novel algorithms that can adapt between the visual and auxiliary sensor features. The first contribution of this chapter is a set of hybrid quality assessment algorithms for UGV, which allows a real-time, NR quality assessment for many multimedia applications. If not stated otherwise, the proposed algorithms can be applied to both independent UGV streams as well as in-parallel recorded video streams capturing the same AoI (needed for video composition).

Most multimedia applications assume a central, high-performance server for the quality assessment [**Shrestha2010**, **Zhang2012**], which limits the assessment's real-time suitability and the scalability. While scalability is a must, resources of the mobile devices are not leveraged in this centralized quality assessment. Thus, the second contribution is a joint selection of appropriate quality assessment algorithms and their optimal placement on processing devices depending on variable application requirements. Applications can specify the timing requirements as well as the minimum precision of the quality metric to the proposed component, which takes care of selecting the best algorithm based on a utility-to-cost ratio. We show that the second contribution can significantly improve scalability and ensures timely quality assessment.

The chapter revises content presented in our peer-reviewed publications [**Wilk2016e**, **Wilk2015c**, **Wilk2016g**].

## 4.1 ARCHITECTURE OF THE QUALITY ASSESSMENT FRAMEWORK

In the proposed scenario, a scalable and adaptive quality assessment is a precondition for quality-aware content adaptation. Novel aspects of this proposed assessment are the respect for varying application requirements, adaptive algorithms, and scalability by leveraging the resources of the mobile devices.

*Varying application requirements* are addressed as multimedia applications (e.g., pursuing video composition) can specify the minimal precision required for an assessment and the maximum runtime until it is completed.

Recording degradations have not yet been analyzed by accurate, objective quality assessment algorithms. Novel recording quality assessment algorithms are proposed, which offer high precision at a low runtime. Proposed algorithms can *adapt* between the signals to be analyzed to achieve given time and precision requirements. Algorithms that can adapt between signals are presented in Figure 20 as *hybrid algorithms*.

*Scalability* is achieved by selecting an appropriate algorithm that complies with the given requirements and dynamically places it on a node in the entire system to efficiently lever-

age the available resources. As a result, this chapter proposes an algorithm selection and placement.

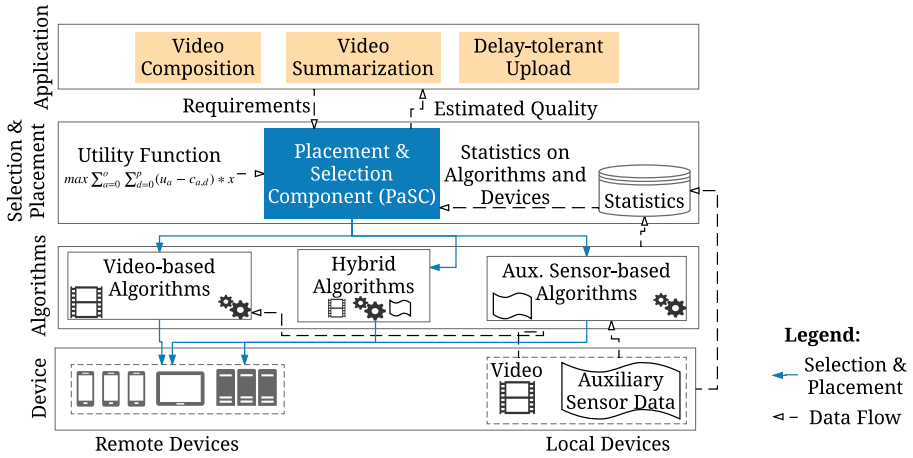



Figure 20: Architecture of the scalable, objective quality assessment

As depicted in Figure 20, a system architecture is proposed to achieve scalability and awareness of application requirements for obtaining an objective quality assessment. An application requires two modifications to integrate the scalable video quality assessment. First, the application needs to specify its requirements for the quality assessment by setting a deadline, when the execution needs to be completed, and the desired precision of the algorithm. Second, the application must be able to receive the quality assessment result.

The Placement and Selection Component (PaSC) is responsible for the selection of a quality assessment algorithm and its placement on a device. All quality assessment algorithms are designed to run on different devices (i.e., see "Device" layer in Figure 20[1]). Whereas a camera, microphones, and other necessary sensors are used on a local device, other devices that run the proposed system are leveraged for algorithm execution. Algorithms are classified into classical video-based and auxiliary sensor-based ones. This classification is applied to algorithms proposed in earlier work (see Section 2.3.3) and the novel algorithms, which are presented in this thesis.

The PaSC also receives the quality assessment result and transmits it to the application. Except for the central repository for statistics, the component including the algorithm definitions is available on each device. As intended, it offers a distributed usage of the system. As depicted in Figure 20, the PaSC uses runtime statistics from a central repository. This repository keeps track of algorithm execution times and stores device statistics on delay, energy, Central Processing Unit (CPU) and memory utilization.

The monitoring of system characteristics happens in an event-driven manner. As soon as the monitored device statistics change, the device pushes updates to the central repository.

## 4.2 RECORDING QUALITY ASSESSMENT ALGORITHMS

The previous section introduced video-based, auxiliary sensor-based, and hybrid algorithms. One major contribution of this work is a first attempt to introduce hybrid algorithms, which adapt and fuse input from different sensors to improve reliability and decrease runtime.

1 The implementation allows the execution of Java-based systems, i.e., Android smartphones and tablets, too. Performance metrics are gathered on a realization of the algorithms in C.

### 4.2.1 *Quality Assessment Stages*

Figure 21 shows the various stages of processing in a quality assessment algorithm. The steps can be classified into

- access of sensors,
- the control stage, which makes an adaptation decision based on sensor samples[2] and synchronizes multi-sensor input,
- the algorithm stage, in which the algorithm processes data from the sensors, and
- the model stage, which generates the quality score.

The different stages of algorithm processing are described in the remainder of this chapter.

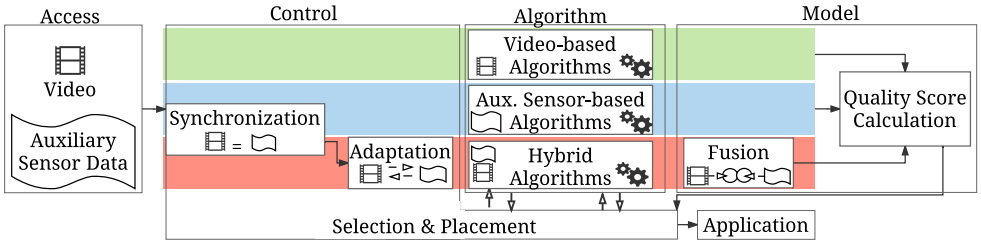



Figure 21: Different processing stages running a quality assessment algorithm

#### 4.2.1.1 *Access*

The access stage is responsible for retrieving sensor input. Sensors can be the camera for video-based algorithms, the microphone of the recording device for an audio analysis, and any other auxiliary sensor in a mobile device, e.g., accelerometer or gyroscope.

As soon as the device offers a new sample, it is received in the access stage. Many algorithms store a series of sensor inputs because they need a temporal assessment. A window of samples is used for $n$ frames in video-based, or $n$ sensor samples in auxiliary sensor-based algorithms. In these cases, the access stage acts as a buffer before invocation of the algorithm stage. Hybrid algorithms access multiple sensors simultaneously and invoke the processing of sensor samples. Note that the synchronization of sensor samples with video frames can be realized by the algorithm or the control stage.

#### 4.2.1.2 *Control*

The control stage is responsible for the coordination of the quality assessment (see Figure 21). It is in charge of synchronizing different sensor inputs, i.e., for auxiliary sensor-based algorithms or hybrid algorithms. As the samples of various sensors are not automatically in sync, synchronization is required to ensure that quality scores that leverage different inputs can be precisely mapped to the media.

Synchronization of the different sensor streams is achieved on a device using the system clock timestamps. During a recording session, drifts in timestamps may occur due to delays of the physical sensors as well as due to the operating system that offers the samples [**Guggenberger2015**]. Related work has shown that during a recording session this

2 This may involve processing of the sensor values in order to make an adaptation decision.

drift may add up to a maximum of 19 milliseconds per hour of recorded videos for recent smartphones such as the LG Nexus 4 and the LG Nexus 5. This drift is negligible, as even for the different modalities of audio and video, a human-perceivable difference occurs at a drift of around $\pm 80$ milliseconds [**Steinmetz1996**].

*Adaptation*

An adaptation between sensors is beneficial if the single sensor algorithms perform differently for different environmental conditions. An adaptation of a hybrid algorithm is applied to either ensure high precision at any time or reduce the runtime at a given precision. Auxiliary sensor signals can be used to determine when environmental conditions are good for applying video-based analysis. For UGV quality assessment, Bano et al. showed that the precision of video-based quality assessment algorithms suffers from significant luminance variations [**Bano2015**]. This finding can be mapped to an adaptation rule to determine if a video-based algorithm is suitable for searching degradations in a video. We determine the sensed lighting around a device for an approximation of the luminance in the video:

$$V_{j,w_i} = \begin{cases} 1, & \text{if } L_{w_i}(j) \geqslant T_{L,D} \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

Here, $V_{j,w_i}$ represents a binary indicator - whether the video-based algorithm shall be considered for a specific video frame $j$ in the sample window $w_i$. $L_{w_i}(j)$ depicts the average light intensity for $j$ in window $w_i$. It is determined by samples from the light sensor in a smartphone. A decision on if a video-based algorithm can be applied for quality assessment is based on whether the majority of the frames in $w_i$ fulfill $V_{j,w_i} = 1$ or not.

$T_{L,D}$ is a device-specific threshold as the sensed samples vary significantly between manufacturers of the light sensors embedded in smartphones [**Bano2015**]. For example, the LG Nexus 4 shows good ambient light conditions at around 100.0 lx, whereas under the same conditions a Samsung Galaxy S2 senses around 20.0 lx. The values were gathered based on the video and sensor dataset of Bano et al. [**Bano2015**].

If a recording device does not offer ambient light sensor values, an image-based algorithm is applied. First, the *RGB* image is converted into a *YUV* representation. Here, Y represents the luma plane and the other two are chrominance (UV) planes. The planes are split, and solely the Y component is used as it depicts the luma intensity. The light intensity per frame is calculated as

$$\mathcal{L}_j = \frac{1}{N_r * N_c} \sum_{k=1}^{N_r} \sum_{l=1}^{N_c} Y_I(k,l) \tag{16}$$

Here, $\mathcal{L}_{w_i}(j)$ represents the average luma intensity of the luma plane ($Y_I$) of frame $j$. $k$ and $l$ represent the pixel coordinates, where $N_r$ gives the height (rows) of a frame and $N_c$ the width (columns). Based on the average luma intensity of a frame, it is determined if an image processing algorithm should be used.

$$V_{j,w_i} = \begin{cases} 1, & \text{if } \mathcal{L}_{w_i}(j) \geqslant T_L \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

Here, $T_L$ is the intensity threshold which ensures a good ambient lighting. In the parameter study of this work, it is shown that $T_L = 50$ offers the best results (see Section 4.4.1.2).

#### 4.2.1.3 *Algorithm Execution*

The algorithm execution stage is the execution environment for different quality assessment algorithms, which differ regarding the sensors used. The algorithms significantly differ regarding precision and runtime. As a platform for running the algorithms the programming language Java with its Runtime Environment is available for both server and mobile devices. Java Native Access (JNA) is used for the invocation of native code to be build specifically for operating systems. The proposed algorithms have been implemented using C, as they are executed significantly faster on mobile devices, e.g., for an LG Nexus 5 between 2.2 to 17.8 times faster[3].

According to their input signal, algorithms are classified into video-based, auxiliary sensor-based and hybrid ones. As they differ in their respective input, they may also differ regarding their results. The model stage is responsible for mapping the algorithm results to a uniform quality value, which can then be used by the application requesting the quality assessment.

#### 4.2.1.4 *Model Stage*

The model stage is responsible for calculating the quality value for all algorithms. A common step for any algorithm represents the mapping of its result to the MOS. The findings of Chapter 3 are used to calculate the MOS for the degradations: camera shake, harmful occlusions, and camera misalignment. The proposed algorithms do not only depict, if a degradation is present, but also quantify their impact; e.g., for a camera shake by depicting the direction, speed, and amplitude of the shake. The calculated result is sent to the application that initially requested the quality assessment.

For hybrid algorithms the model stage is also responsible for fusing results provided by subalgorithms. Hybrid algorithms analyze inputs from different sensors in different subalgorithms. Instead of relying on only one sensor, the combination of results derived from multiple sensor inputs can improve the precision [**Cricri2012**]. An algorithm designer can decide when to calculate a fused score $S_{V,AS}$:

$$S_{V,AS} = o_V \times S_{A_V,w_{i,V}} + o_{AS} \times S_{A_{AS},w_{i,AS}} \tag{18}$$

Here, $o_V$ represents the weight for the degradation score calculated by the video-based algorithm, and $o_{AS}$ the score based on auxiliary sensors. The weights $o_V$ and $o_{AS}$ are normalized between $[0, 1]$, and the condition $o_V + o_{AS} = 1$ must hold. $w_{i,AS}$ and $w_{i,V}$ represent the auxiliary sensor sample window and the video frame window used in the algorithm execution stage. Algorithms use the fusion of the algorithm output when subalgorithms present contradicting results.

In the remaining subsections, algorithms for detecting and quantifying degradations are given for camera shakes, harmful occlusions, and camera misalignments.

### 4.2.2 *Camera Shake Assessment*

Chapter 3 has shown that camera shakes reduce the perceived quality of a video. As no reliable algorithm exists to quantify its impact, it is proposed to have an auxiliary sensor-based algorithm, a video-based algorithm, and a hybrid algorithm combining the two.

3 http://www.learnopengles.com/a-performance-comparison-between-java-and-c-on-the-nexus-5/; Visited on: 07/20/2016

#### 4.2.2.1 *Auxiliary Sensor-based Algorithm*

The identification of camera shakes during recording is based on the three-dimensional linear accelerometer sensor. It offers a fine-grained measurement of the acceleration without gravity components in $\frac{m}{s^2}$ on three axes. For the camera shake assessment, a window of $m$ samples is used to monitor the device motion. The definition of a small sample window is based on the work on camera tilts and panning by Cricri et al. [**Cricri2012**]. To reduce the computational burden, the linear accelerometer gathers a subset of 15 samples ($f = 15$ Hz) in a window of one second. A low-pass filter is applied to reduce the window size and avoid noise generated by the sensitive sensors[4]. In contrast to camera tilts and pans, shakes can be identified by at least two consecutive direction changes. The number of direction changes is measured by the counter $c_d$. If no movement is measured for the sample size $m$, the counter $c_d$ is reset. The resulting algorithm (see Algorithm 1) illustrates how a camera shake can be detected based on direction changes.

**Algorithm 1** Proposed algorithm for the detection of camera shakes based on the linear accelerometer.

**function sgn**(p,q): 3D-Signum-Function - calculates the sign of the difference of p to q for each dimension of a 3D vector and returns a 3D vector.
**function MAX**(p[]): Calculates highest absolute acceleration in the array p[] of three-dimensional vectors
**Require:** s[]: Three-dimensional sample array (x,y,z) of size $m$ filled with linear accelerometer sensor samples
**Require:** t: Latest sample index
**Require:** $T_{S,AS}$: Threshold for identifying significant camera shake
$c_d \leftarrow 0$
**if** $s_{t,x} \neq s_{t-1,x}$ **or** $s_{t,y} \neq s_{t-1,y}$ **or** $s_{t,z} \neq s_{t-1,z}$ **then**
  **for** $i \leftarrow 0..t-3$ **do**
    **if** $sgn(s_i, s_{i+1}) \neq sgn(s_{i+1}, s_{i+2})$ **then**
      $c_d \leftarrow c_d + 1$
    **end if**
  **end for**
  **if** $c_d \geqslant 2$ **and** $max(s[]) \geqslant T_{S,AS}$ **then**
    **return** *true*
  **end if**
**end if**
**return** *false*

A detected camera shake in a window of $m$ samples is used to determine not only if a shake is present but also to which extent it degrades the perceived quality. The speed (frequency) and amplitude of camera shakes are calculated on the basis of the directional changes. The algorithm applies a signum function to determine the direction of linear accelerometer values. To compensate small and imperceivable changes, a threshold $T_{S,AS} = 0.2\frac{m}{s^2}$ determines the minimum acceleration for a harmful shake.

Besides this classification, a quantification is also possible solely relying on $\frac{c_d}{l(w_i)}\frac{1}{s}$, by calculating the frequency of the shake. Here, $l$ depicts a length function for retrieving the window size and $c_d$ gives the counter of direction changes. The amplitude of a camera shake is determined by the time a camera movement into a specific direction is

4 The low-pass filter uses a threshold of $\alpha = 0.3$ and iterates linear accelerometer samples $a$ with index $i$ by applying $a_i = a_{i-1} + \alpha * (a_i - a_{i-1})$.

detected by measuring both a dense window of samples from the linear accelerometer ($> \frac{16}{s}$) and the corresponding timestamps for the readings. The sample window is iterated to compute the average acceleration in each direction until a direction change is detected. Then the distance into a specific direction is computed as $D = \Delta t^2 * \overline{LA(x,y,z)}[m]$, where $\Delta t$ represents a continuous movement about a specific axis measured in [s] and $\overline{LA(x,y,z)}$ depicts the linear accelerometer values[5]. A mapping is still needed for the camera model to map the distance in meters to pixels that were crossed in a given window. This mapping is based on an initial calibration step, e.g., when a new device runs the quality assessment for the first time. A combined analysis of the correlation of different linear accelerometer samples to pixel movements is performed using a Canny filter [**Canny1986**]. After this initial calibration and synchronization step, no video-based techniques need to be applied. The determined influence factors are then mapped to the formula known from Chapter 3 to determine the Differential Mean Opinion Score (DMOS) as $\Delta MOS = 0.8739 * x_{amplitude} + 0.0682 * x_{time} + 2.961 * x_{speed}$, or the respective genre-dependent coefficients. The DMOS determines the absolute quality loss induced by a camera shake.

#### 4.2.2.2 *Video-based Algorithm*

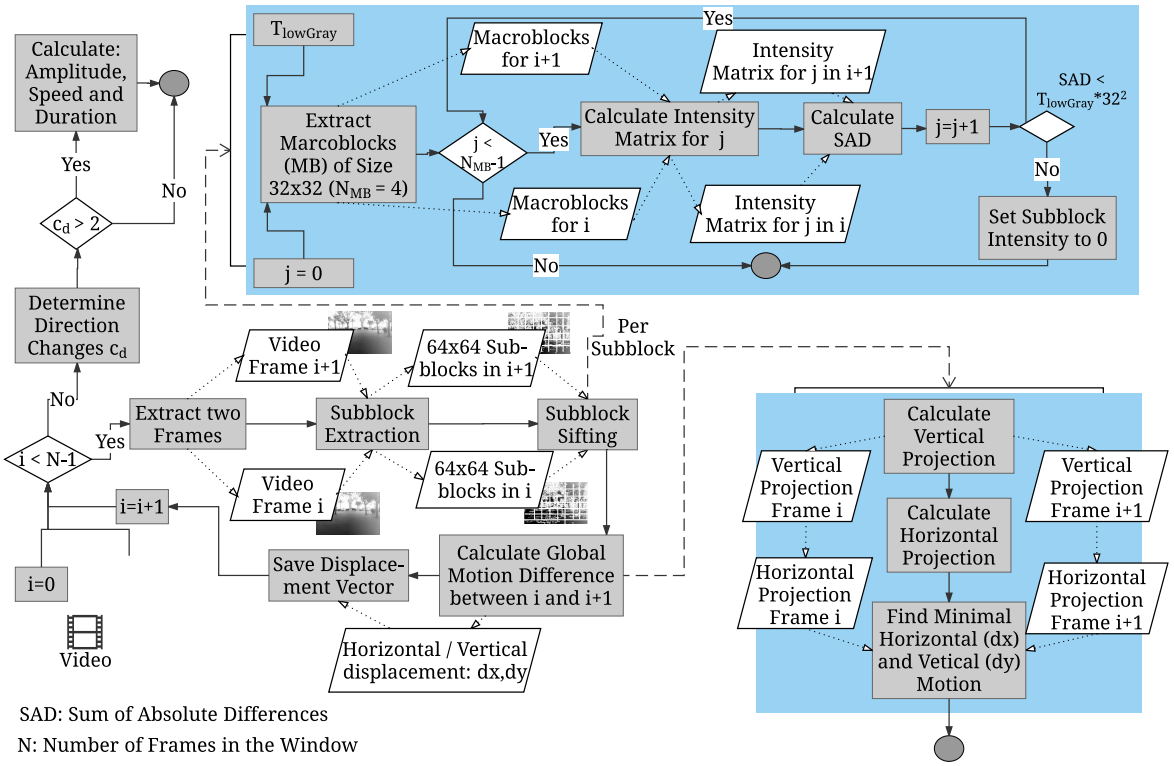
The video-based algorithm for camera shake detection performs a global motion analysis between the two video frames represented as intensity matrices. Based on this analysis, repetitive changes in motion are classified as camera shakes. Figure 22 gives an overview of the algorithm. In a window of N video frames, the algorithm extracts two consecutive video frames per iteration. A fast intensity-based approach is used that is based on the Fast Subblock Gray Projection Algorithm (FSGPA) [**Ha02013**]. For detecting motion, $64 \times 64$ pixel subblocks of the intensity representations are extracted from each video frame. The subblock detection or sifting step filters subblocks whose average intensity per pixel is below $T_{lowGray} = 98$ (see Section 4.4.1.2). All subblocks with such a low-intensity are not considered for motion calculation. The rationale behind this step is that motion cannot be reliably determined in low-intensity frames.

The global motion between two filtered frames is estimated by analyzing the horizontal and vertical projections of a video frame independent of each other. Vertical and horizontal motion vectors are determined by finding the minimal, common movement of all subblocks - indicated by a minimal error when mapping the two intensity frames. In a search breadth of $m = 1$ for each subblock, it is determined whether it has moved up to $2 * m + 1$ blocks. The minimum of the Sum of Absolute Differences (SAD) indicates the steps a subblock has moved, i.e., in the horizontal and vertical translation:

$$d_x = m + 1 - w_{min,h},\ d_y = m + 1 - w_{min,v} \tag{19}$$

Here, $w_{min,h}$ indicates an alignment to compensate for motion in the horizontal direction. Similarly, $w_{min,v}$ indicates the vertical alignment. Based on this motion estimation, the number of direction changes can be determined. The motion estimation is recalculated for all entries in a window of video frames. As soon as at least two direction changes are detected, a camera shake is identified. Using methods proposed by Saini et al. and Campanella et al., pan, tilt, and shake can be distinguished using a median or low-pass filter on the motion vectors $d_x$ and $d_y$ [**Campanella2007**, **Saini2012**]. The SAD is computed between the original and filtered motion vectors. The absolute differences of the intensity

5 A calibration of the device is needed with approximately 100 samples to reliably determine the individual white noise of gyroscope and accelerometer.




SAD: Sum of Absolute Differences

N: Number of Frames in the Window

Figure 22: Flow chart of the camera shake detection algorithm, relying on a video-based detection of motion in the horizontal and vertical plane. The blue boxes detail essential steps in the process. Pictures illustrating video frames are taken from the *CMDG* dataset [**Bano2015**].

values are summed to determine a shake score in the range of $[0, 1]$. $S_{S,V}$ represents a shake score that is determined based on the calculated SAD in the vertical and horizontal projection of the video frames:

$$S_{S,V} = \sqrt{SAD_h{}^2 + SAD_v{}^2} \tag{20}$$

Based on the $T_{S,V}$ a shake can be distinguished from an intended pan and tilt. If the detected score is above this threshold, the motion is classified as shake. This condition triggers the calculation of the amplitude, speed, and duration of a camera shake. Similar to the auxiliary sensor-based camera shake detection, the findings of our subjective analysis can be used to determine the degradation impact by calculating the DMOS as $\Delta MOS = 0.8739 * a_{ampl} + 0.0682 * a_{dur} + 2.961 * a_{speed}$.

### 4.2.3 *Harmful Occlusion*

Video-based algorithms are proposed to detect and assess harmful occlusions. Additionally, a contribution towards the design of adaptive video-based algorithms for harmful occlusion detection in UGV is made. Auxiliary sensors are used to determine when to switch between the video-based algorithms.

#### 4.2.3.1 *Edge Density-based Occlusion Detection*

The proposed approach extends the research of Saini et al., who proposed to calculate the edge density of video frame patches [**Saini2012**]. A harmful occlusion requires that an occluding object must be closer to the camera than the recorded AoI. In a two-dimensional

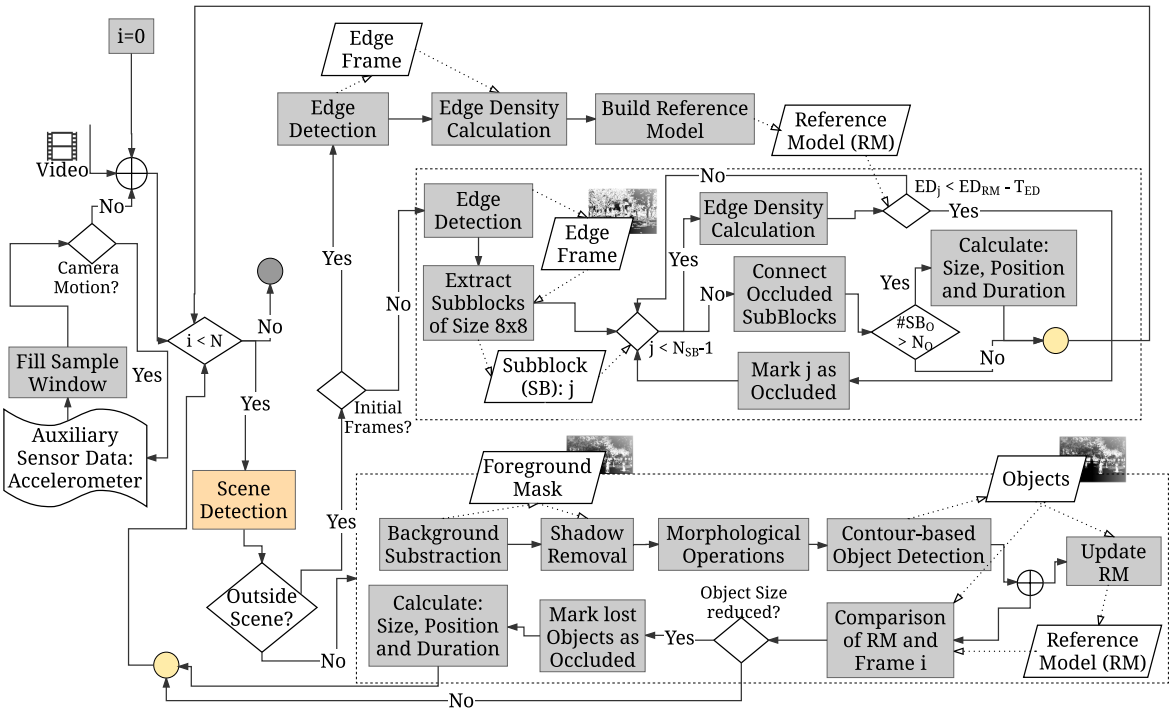



Figure 23: Overview of the proposed occlusion detection algorithm showing the video-based calculation and its auxiliary sensor-based adaptation.

video frame representation, an object being recorded at a close distance to the camera lense has a higher average distance between the object's contour pixels than the contour pixels of the same object at a larger distance to the camera. When tracking a pixel block in a video over time, a sudden increase of the edge distances indicates a possible occlusion.

The process of detecting these occlusions is shown in Figure 23, which calculates an edge density for consecutive video frames in different blocks of size $8 \times 8$. After detecting edges using the Canny edge filter [**Canny1986**], the edge density is calculated using the edge frame and convoluting it with a $3 \times 3$ unity matrix of ones. This step strengthens edges present in the video frame.

The edge pixels are counted per subblock:

$$ED_j = \sum_{i=1}^{N_r} \sum_{k=1}^{N_c} SB_I(i,k) \tag{21}$$

Here, $j$ iterates over all subblocks of size $8 \times 8$, and $i$ and $k$ depict the indices for the subblock pixels. For each subblock $j$, the edge density $ED$ is calculated by summing the intensity $SB_I$. As soon as a block's edge density $ED_j$ falls below $ED_{RM} - T_{ED}$, the subblock is labeled as being harmfully occluded. This threshold helps to separate structures with a low-edge density from harmful occlusion candidates as the edge density must significantly drop from a reference value. These objects, containing low-edge densities, would otherwise be classified as harmful occlusions. Also, to fill small gaps in occluded areas, a connected component analysis is performed. Each non-occluded subblock is labeled as being occluded, if all neighboring subblocks indicate an occlusion.

#### 4.2.3.2 *Object Tracking-based Occlusion Detection*

The second algorithm applies occlusion detection by tracking foreground objects. It relies on a background-to-foreground segmentation. A foreground mask detection algorithm

proposed by Zivkovic et al. is used and improved for UGV with a shadow removal technique applied as proposed by Saravanakumar et al. to remove contours from the video frame which do not belong to objects being tracked [**Saravanakumar2012**, **Zivkovic2006**]. This approach significantly reduces the false classification rate. It applies a normalized cross-covariance calculation to the frame to detect shadows. If shadow candidates reach a normalized cross-covariance of 50,[6] they are classified as shadows and are removed from the frame. This approach requires that a foreground model exists, which is achieved by using Zivkovic et al.'s approach [**Zivkovic2006**]. Background objects in a certain distance are smaller than objects closer to the recording camera. Thus, when applying a contour filter to a frame, the area with background objects is assumed to be small in comparison to foreground objects. The results of applying the shadow and background removal are depicted in Figure 24.

(a) Original frame with occlusion

(b) Foreground mask by frame difference method

(c) Original frame with occlusion

(d) Foreground mask

Figure 24: Foreground mask detection by the frame difference method [**Saravanakumar2012**].

The remaining foreground frame is compared to a per-frame updated reference model (contours). From the contours, independent objects are detected and counted. An occlusion is assumed, if in the reference model foreground objects cannot be tracked anymore.

The position, size, and duration of an occlusion can then be computed for both approaches in a similar manner and applied to the MOS as described in Chapter 3. The ratio of patches occluded in relation to all patches is then used as the size of the occlusion. The perceived quality reduction (DMOS) is calculated based on the quality model proposed in Section 3.3.

In comparison to the other algorithms presented in this chapter, the harmful occlusion detection is computational intensive as all subalgorithms rely on video analysis. The video composition application described in Chapter 6 requires that live video streams are analyzed for harmful occlusion. In Section 4.4 we show that on high-end servers the proposed hybrid algorithm can be processed in real-time. Existing smart mobile devices are not suitable for a timely harmful occlusion detection in live video streams.

6 Determined as an optimal threshold in our parameter study.

#### 4.2.3.3 *Auxiliary Sensor-based Control (Adaptation)*

Adaptation is proposed between the edge-density and the object-tracking algorithm, as lighting conditions and video motion affect the reliability of an algorithm's performance. Whereas the tracking-based algorithm has shown a certain robustness against small but rapid luma changes, the edge density-based approach requires constant brightness and high ambient lighting.

For applying the object tracking-based approach, stable recording conditions are needed, such as no camera motion and a known orientation of the recording device. An auxiliary sensor-based adaptation based on the linear accelerometer is applied. The tracking-based occlusion detection is only invoked under stable conditions, so there is no movement of the camera. Also, the auxiliary sensors are used to avoid generating reference models for the edge density-based algorithm under motion. Motion leads to blur in the video frames, which would lead to a significant deviation from the reference edge density values in a frame. For detecting stable conditions the threshold $T_{S,AS}$ is used, which determines a significant motion in the camera shake algorithm. In a sample window $s[]$, the condition $max(s[]) \geqslant T_{S,AS}$ indicates a significant motion.

### 4.2.4 *Camera Misalignment and Tilt*

Remaining camera degradations discussed in this thesis are categorized into (1) the camera tilt detection: the detection if a camera is rotated about its z-axis and (2) the detection of camera misalignment as discussed in Section 3.1. To address (1) the camera tilt detection an auxiliary sensor-based algorithm, we propose a video-based algorithm and an adaptation between the two. As explained in Section 4.2.1.2, an adaptation is performed to circumvent bad lighting conditions that could affect the results of a video-based algorithm. The respective video-based algorithm is shown in Figure 25.

For (2), the detection of a camera misalignment as discussed in Section 3.1, the commonly recorded scene (AoI) is determined by leveraging an auxiliary sensor-based algorithm.

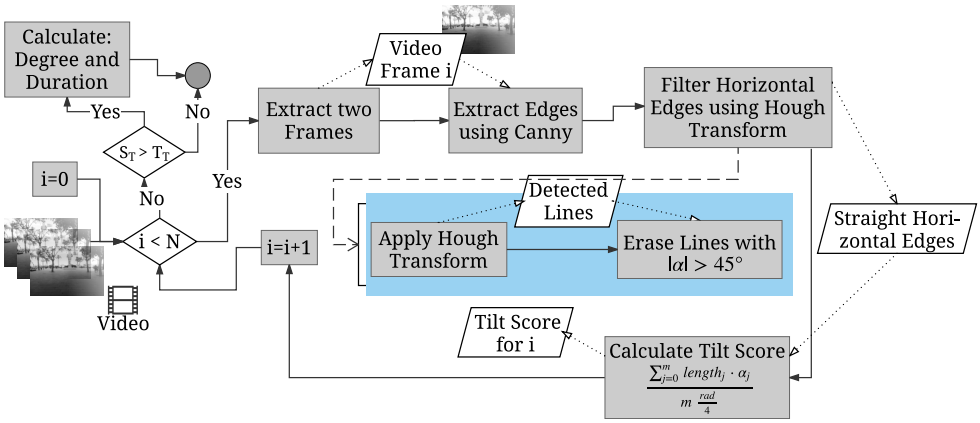



Figure 25: Video-based camera tilt detection algorithm.

#### 4.2.4.1 *Video-based Algorithm*

The algorithm as depicted in Figure 25 detects tilt in video recordings. The algorithm extracts horizontal edges - or vertical ones in the case of a portrait-oriented recording - based on the Canny edge filter **[Canny1986]**, and determines straight lines using a Hough

transform. The Hough transform reduces the detected lines to those which are straight lines as contours of rectangular objects. Furthermore, straight horizontal lines are required, as only those allow for a reliable determination of an angle of the tilt to the horizontal plane. In comparison to related approaches we do not require that the majority of edges in a video frame are horizontal edges [**Saini2012**]. Also, straight lines with an angle greater than 45° will be erased as they would misleadingly indicate a wrong orientation detected by the device. In summary, straight horizontal lines do not need to be perfectly aligned with the horizontal plane, but should have an angle of less or equal 45° towards the plane. Afterwards, an exact direction for each line is determined. In a final step, the tilt score (which is normalized $[0, 1]$) is calculated by applying a calculation of the angle of a detected line to a perfect horizontal line as $\alpha$. This angle is multiplied by the number of pixels of all horizontal lines and averaged. Here, a measure close to 1 indicates long lines with a significantly tilted angle. A threshold determines whether a tilt degrading the perceived quality is detected.

#### 4.2.4.2 *Auxiliary Sensor-based Algorithm*

*Tilt Detection*

An auxiliary sensor-based approach is proposed which relies on the accelerometer[7] in smart mobile devices. The algorithm is straight-forward and leverages the x- ($a_x$) and y-components ($a_y$) of the acceleration to determine the angle the y-axis of the smart mobile device is rotated. This angle determines the deviation to an undistorted reference.

A reference model is built to determine the initial orientation of the device ($O_D$). We determine the tilt in relation to this initial $O_D$. Based on the difference of the global orientation of a device, the tilt is calculated as

$$\beta_{Tilt} = |\alpha_{Init} - \alpha| \tag{22}$$

where $\alpha$ is determined by $\alpha = \arctan(\frac{a_x}{a_y})$. $\alpha_{Init} = \arctan(\frac{a_x}{a_y})$ is calculated in the first seconds of a video recording. We assume that these initial seconds of a video are captured without a degradation.

A drawback of the accelerometer is its sensitivity to small movements and white noise. To improve the robustness, $\alpha_{Init}$ is calculated again after the recording device has moved. During a movement the hybrid algorithm does not use the auxiliary sensor-based subalgorithm but adapts to the video-based calculation. Also, a sample window of two seconds at a sampling rate of 4 Hz is used and averaged for calculating the tilt angle.

A tilt that degrades the perceived quality is determined by $\beta_{Tilt}$ being larger than $T_{CM,AS}$. $T_{CM,AS}$ represents the threshold that distinguishes tilts being transparent to the viewer and tilts representing a major degradation.

*Collaborative Detection of Camera Misalignments*

To detect camera misalignment, we propose a collaborative sensing approach. This approach is only applicable, if multiple recording devices record the same AoI. This scenario is illustrated in Figure 26. Here, all users except the red one are recording the AoI. We assume that the red user is recording an uninteresting part of a scene. As a simplification, the three-dimensional world is simplified to a two-dimensional model. A collaborative

7 The algorithm can be applied to the gyroscope, too. The reduced noise in the gyroscope samples and the higher sampling rate are not required for tilt detection. An disadvantage of the gyroscope is the higher energy footprint [**Koenig2013**].

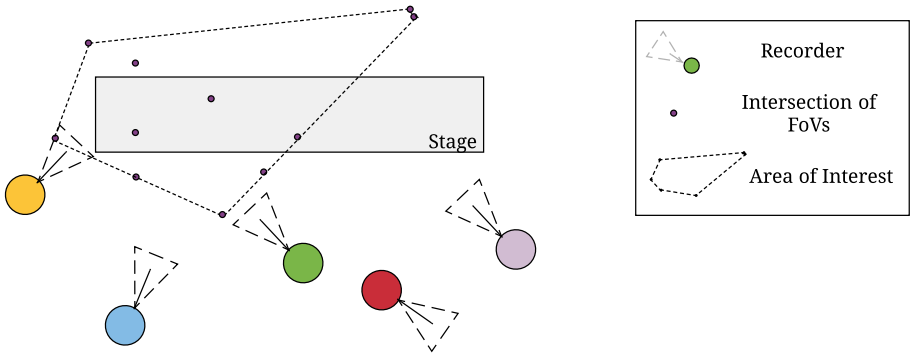



Figure 26: Collaborative sensing approach for eliminating views that do not record the AoI of an event.

sensing approach is chosen to detect the AoI. Our aim is to identify users who record non-interesting part of an event. The decision is solely based on the orientation and the FoV of each recorder. Based on FoV border lines, the intersection points of the FoVs of all recording devices can be calculated. Majority voting is conducted on the most inner-lying intersection points of FoV border lines, which are in the viewing direction of the recording devices. The FoV of each smart mobile device can be calculated based on the focal length $L_F$ as $FoV = 2 \times arctan(\frac{wi_{AS}}{2 \times L_F})$, where $wi_{AS}$ represents the horizontal width of the sensor plane. Both parameters $L_F$ and the effective focal length $wi_{AS}$ are measured from EXIF tags from a captured photo on a smartphone. Also, the Android OS offers an Application Programming Interface (API) to retrieve the FoV from camera parameters.

For detecting the AoI, a convex hull of the intersection points is applied where in-lying points are erased[8]. For a quick calculation of the convex hull, the quick elimination approach ($O = N$) is used, which chooses a rectangle or quadrilateral of four points in the point set. Intersection points that are obviously non-feasible (large distance) are discarded upfront. Building the convex hull is achieved by using a Graham Scan **[Graham1972]**, which relies on a polar sorted list of points. It starts at the lowest vertical point and systematically erases points which would result in a clockwise turn.

From the FoVs of the recording devices, those video streams can be determined which do not record the common AoI. Furthermore, the deviation from the center of the AoI is calculated, which allows to determine the characteristics of the quality model, as proposed in Section 3.3.

## 4.3 JOINT SELECTION AND PLACEMENT OF ALGORITHMS

Besides novel algorithms for the detection and assessment of degradations common in UGV, the assessment of video streams at runtime is a huge burden for centralized services. A scalable solution is proposed, which leverages the resources of all recording devices and dynamically allocates quality assessment processes to the most appropriate devices. A joint selection of an appropriate quality assessment algorithm and the placement of this algorithm on a device is proposed.

Mobile video broadcasting includes several devices for the quality assessment. The first node is the mobile device recording the stream. Also, all close-by devices participate in the mobile broadcasting service can be used for running a quality assessment. Besides the

8 The basis for the calculations is the location data gathered from the GPS that is mapped into a two-dimensional coordinate system based on the UTM.

mobile devices, the receiving server, as well as other networking elements, can potentially run quality assessment algorithms.

The selection determines which algorithm meets given application requirements regarding precision and runtime. These algorithms are classified into assessment types (AT), which not only include the recording quality assessment algorithms (proposed in the previous section), but also traditional video quality assessment metrics, that measure the effects of transport artifacts or compression.

### 4.3.1 *The Placement and Selection Component*

Figure 20 illustrates an overview on the PaSC. A common, decoupled component is available for the execution of the algorithms and the selection and placement decision. It con-

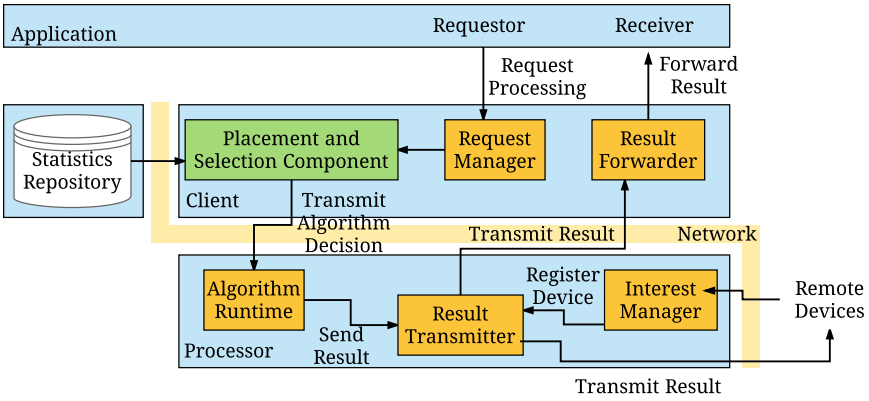



Figure 27: Subcomponents of the PaSC running on each smart mobile device for using the proposed scalable and adaptive quality assessment.

sists of the following major building blocks: a statistics repository, an algorithm processor, and the algorithm client. Applications request the processing of algorithms and receives the results after completion. A client consists of the respective components to manage requests for algorithm execution from an application and the forwarding of the algorithm execution results. The core element of the client is the ”Placement and Selection” component. As it is integrated into the client, each device individually makes the placement and selection decisions. Decision-making is based on statistics that devices regularly request from the repository. Processing of an algorithm happens on the devices (local processing), or relevant data is transmitted for remote processing to either other mobile devices or a server. In case a local device is selected, the computation can be started immediately on the device without any data transmission. For remote processing, a device receives the required sensor readings, e.g., the video. The respective components of the processor include the algorithm execution environment necessary for running the quality assessment algorithm, and components required for the result distribution. Multiple devices, e.g., the local device and the server, can be interested in the quality assessment result. Thus, a quality assessment request is handed to the interest manager module which allows any device to register for assessment results of a specific video stream. Once the result is available, all interested devices are informed. A mobile device may request a video quality assessment, and a composition server is informed about this assessment result. After each execution of an algorithm on a device, the repository is informed of the current device state and statistics - including available and used energy, the runtime of the algorithm, and the avail-

able and used memory of the device. Coordination of statistics is achieved by using the repository block that builds a shared storage for statistics.

### 4.3.2 *Steps for Selecting an Algorithm and a Processing Device*

The steps of selecting an algorithm and placing it on a device are illustrated in Figure 28. Once an application requests a quality assessment of a video stream, the device recording the video requests a list of the devices available for processing and the related statistics from the repository.

All available devices and the subset of the algorithms are then considered in the selection, as described in Section 4.3.3.

A preprocessing step allows algorithms to determine their runtime characteristics on any device. Usually, this preprocessing requires the execution of algorithms on a device before it runs the PaSC, which is unfeasible in a real deployment. Thus, devices are categorized into "device types" to reduce the preprocessing effort. A "device type" can be the specific model of a smart mobile device; or, as in our case, a categorization according to high-end servers, low-end, medium-end and high-end smartphones. A device is classified when it starts using the PaSC. The runtime measurements of other devices of the same "device type" can then be used for a joining device. In this thesis, the devices are classified using the results of the benchmark databases Vellamo[9] and AnTuTu benchmarks[10] (see Section 4.4.2.1).

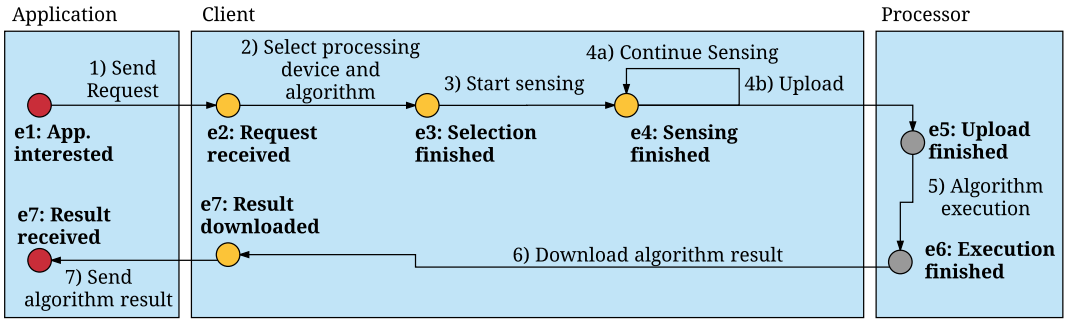



Figure 28: PaSC: From sensor values to the measured results.

Figure 28 depicts the steps and occurring events in this component for requesting (e1), selection, and placement of an algorithm (e3 - e5), and the processing of the algorithm (e6). The selection and placement decision-making are explained in the next section.

### 4.3.3 *Selection and Placement Algorithm*

The goal is to select one algorithm $a \in S_{AT}(a_{AT,1}...a_{AT,n})$ to be processed by a device $d \in D(d_1...d_m)$. $S_{AT}$ represents the subset of algorithms that analyze a specific set of degradations, i.e., recording quality, video quality, or audio quality assessment. The selection of an algorithm and a processing device relies on a comparison of the utility $u_a$

9 Qualcomm Vellamo Metal Benchmark; Visited on: 10/09/2016
10 AnTuTu Benchmark, http://www.antutu.com/en/index.shtml; Visited on: 10/09/2016

and the costs $c_{a,d}$ for processing $a$ on $d$. For all algorithms and all devices, the idea is to maximize the difference of utility ($u_a$) to costs ($c_{a,d}$):

$$max \sum_{a=0}^{o} \sum_{d=0}^{p} (u_a - c_{a,d}) * x_{a,d} \tag{23}$$

It determines the maximum utility-to-cost proportion for executing an algorithm $a$ on a device $d$. The result is in the range $[-1, 1]$ as both $u_a$ and $c_{a,d}$ are normalized to $[0, 1]$.

To ensure that exactly one algorithm $a$ is processed on one device $d$ the following condition must hold:

$$\sum_{a=0}^{o} \sum_{d=0}^{p} x_{a,d} = 1 \tag{24}$$

Here, $x_{a,d}$, a binary value, shows if $a$ is executed on $d$ where $x_{a,d} \in \{0, 1\}$. Practically speaking, we iterate over all combinations of running each algorithm on any device and determine the best combination.

The utility ($u_a$) is represented by the precision of an algorithm and is described in our case as its precision. The metric is device-independent as we assume deterministic algorithms. The cost statistics include the *processing time* ($\tilde{t}(a, d)$), *energy* ($\tilde{e}(a, d)$), *memory* ($\tilde{m}(a, d)$), and *data traffic* ($\tilde{dt}(a, d)$) spent by an algorithm. The costs $c(a, d)$ are calculated by:

$$w_t * \frac{\tilde{t}(a, d)}{N_c} + w_e * \frac{\tilde{e}(a, d)}{N_c} + w_m * \frac{\tilde{m}(a, d)}{N_c} + w_{dt} * \frac{\tilde{dt}(a, d)}{N_c} \tag{25}$$

Furthermore, $c(a, d)$ uses $N_c = 4$ to normalize results to a range between $[0, 1]$. $N_c$ represents the number of different cost components of $c(a, d)$. Both utility ($u_a$) and costs ($c(a, d)$) are normalized to $[0, 1]$. Here, we define the $\tilde{t}(a, d)$ to be the normalized scalar in respect to the maximum runtime measured for an algorithm $a$ on any available device in $D(d_1 ... d_m)$. For the execution time it results in:

$$\tilde{t}(a, d) = \frac{\overline{t(a, d)}}{t(a_{AT,max})} \tag{26}$$

$\tilde{t}(a, d)$ is calculated as the average of all completed runs of algorithm $a$ on a device $d$ in comparison to the maximum runtime ($t(a_{AT,max})$) of algorithms in $S_{AT}$ and the device set D. $t(a_{AT,max})$ represents the maximum runtime of the slowest running algorithm in the set $S_{AT}$: $a_{AT,max} = argmax_a(t(a, d))$. The remaining cost components have similar formulas and are thus not discussed in detail.

The costs can be adjusted by the weights $w_t$, $w_e$, $w_m$ and $w_{dt}$, which indicate the relevance of different cost components where a value of zero represents no relevance and a value of 1 represents the maximum relevance for the selection of an algorithm: $w_t \in [0, 1]$, $w_e \in [0, 1]$, $w_m \in [0, 1]$ and $w_{dt} \in [0, 1]$. These weights can be set by application designers to adjust the impact of certain cost components on the selection. E.g., for applications where the caused data traffic is rather negligible the weight $w_{dt}$ can be set close to 0. For the proposed video composition scenarios, the weights are set according to $w_t = 1$, $w_e = 1$, $w_m = 0.5$ and $w_{dt} = 0.5$ in order to favor algorithms with short processing times in the selection process.

#### 4.3.4 *Algorithm Implementation to Support the PaSC*

PaSC is available for the Java Runtime Environment and Android[11]. Thus, all algorithms are realized using Java (auxiliary-sensor based algorithms) or are encapsulated by the Java Native Code injecting C code (video-based algorithms). Video-based algorithms leverage JNA as the algorithms implemented using C run significantly faster for mobile devices. The speed-up for a LG Nexus 5 is between 2.2 to 17.8 times. The PaSC assumes that the algorithm implementations are available on all devices. The integration of a new algorithm requires an update of the PaSC. A solution to this issue, which is not investigated in this thesis, is the migration of code at runtime. Related research provides frameworks that can be used to support code migration [**Aitenbichler2007**].

Each device reports the execution time, a sample on the used device memory, a measurement of the energy drain, and an estimation on the caused data traffic after an algorithm run is completed. The energy drain is only measured for mobile devices. We leverage the implementation of the PowerTutor tool provided by the University of Michigan[12] to estimate the energy drain caused by an algorithm execution. The usage of PowerTutor for measuring the energy consumption of video-based algorithm has shown to provide a sufficient reliability [**Ganiyu2012**]. The remaining statistics (processing time, data traffic and memory) are implemented using Java and OS functions.

## 4.4 SETUP OF THE EVALUATION

The analysis of the precision and runtime behavior of the novel quality assessment algorithms (Section 4.4.1) is split from the analysis of the PaSC (Section 4.4.2).

### 4.4.1 *Recording Quality Assessment*

#### 4.4.1.1 *Evaluation Setup for the Recording Quality Assessment Algorithms*

The evaluation executes recording quality assessment algorithms and compares them on different, publicly available video datasets in comparison to state-of-the-art algorithms. Furthermore, results are given regarding the reliability of the quality assessment prediction and its runtime. Metrics are introduced depicting these attributes. The algorithms have been evaluated on an Intel Xeon CPU E5-1650 with 64 GB of dedicated memory, without GPU support.

*Video Datasets*

For assessing the proposed algorithms detecting camera shakes, harmful occlusions and camera misalignments datasets are used that contain UGV and auxiliary sensor data including gyroscope, accelerometer, and location samples.

The CMDG dataset [**Ban02015**] contains 24 UGVs with a total duration of 70 minutes at a resolution of 480p up to 1080p, and a frame rate of 23 FPS at different brightness levels. Videos are annotated by synchronized samples of the gyroscope, GPS, and accelerometer sensors. The dataset contains a manually annotated ground truth for camera shake detection, but lacks annotations on harmful occlusions or camera misalignments.

11 Details on the implementation are available from our repository Chapter A.
12 http://ziyang.eecs.umich.edu/projects/powertutor/; Visited on: 10/13/2016

The second public dataset used for evaluation is called JIKU, provided by Saini et al. **[Saini2012]**. The dataset contains UGVs from ten smartphones with 66 video clips and a total duration of 343 minutes. Four recordings have a resolution of 480p, whereas the remaining videos have a resolution of 720p at a frame rate of 25 FPS. The videos are annotated by readings from the sensors compass and accelerometer. A ground truth had to be created for the recording degradations of camera shake, occlusion, and camera misalignment.

For the parameter study and the evaluation, one additional video dataset was created that consisted of 18 UGVs - including high and low brightness frames. Nine videos were recorded at a resolution of 1024x480 (480p) by a Nexus 5 and annotated by the accelerometer, location or gyroscope samples. Nine additional videos have a resolution of 720p and a duration between $1-2$ minutes at a frame rate of 29 FPS. The ground truth is manually annotated for camera shake, occlusion and camera misalignment. The recordings in this dataset were purposely created for algorithms to easily find degradations. Also, the recordings lack other degradations such as compression artifacts or rapid motion of objects. In the remaining evaluation, this dataset is called UGVD.

*Evaluation Metrics*

Metrics used in the evaluation describe the runtime of the algorithms and their ability to reliably detect a specific degradation.

The total time of computation for a hybrid algorithm is as follows:

$$t_{AL}=t_{AD}+t_{AS}+t_V \qquad [s] \tag{27}$$

$t_{AL}$ depicts the entire runtime in seconds for an algorithm, where $t_{AD}$ depicts the time necessary for making an adaptation decision in hybrid algorithms, $t_{AS}$ the runtime of the auxiliary sensor-based analysis, and $t_V$ the video-based processing part. If an algorithm is evaluated that solely uses video as input, $t_{AD}=0$ and $t_{AS}=0$. Derived from the runtime of an algorithm, real-time suitability is calculated as a fraction of $t_{AL}$ in respect of the video duration: $RTS=\frac{t_{AL}}{D_V}$.

To determine the precision of algorithms, standard metrics are used that compare the quality assessment result with the ground truth for each dataset. Based on this comparison, the numbers of true positive (TP), false positive (FP), true negatives (TN), and false negative (FN) detections are calculated. As a result, the metrics of precision, accuracy, recall, and the F1-score are used. The precision depicts the rate of truly detected degradations of an algorithm in relation to the total number of detections in a video: $P=\frac{TP}{TP+FP}$. Recall is the number of true detections made by the algorithm in relation to true positives and degradations missed by the algorithm, which can be described as $R=\frac{TP}{TP+FN}$. Thus, the precision depicts how many degradations were found in a video that are not degrading the quality as annotated in the ground truth of the dataset; the recall describes the missed degradations. Both concepts are fused in a single metric, the F1-score as $F1=2\times\frac{P\times R}{P+R}$.

#### 4.4.1.2 *Parameter Study*

In a first step of the evaluation, the algorithms' parameters are determined which achieve the highest F1-score, thus performing best. The parameters determined are used in the remaining evaluation.

The study was performed on a seven video subset of the JIKU dataset[13]. Also, due to the absence of light sensor values in the JIKU dataset, a subset of videos from the CMDG dataset is selected[14] for empirical studies.

*Parameters for Camera Shake Assessment*

Table 8 gives an overview of the parameters evaluated for camera shake assessment. It includes both video-based and auxiliary sensor-based (linear accelerometer) parameters. Values annotated by an * give the optimal F1 score.

| Parameter | Description | Unit | Parameter Variations |
|---|---|---|---|
| Adaptation | | | |
| $T_{L,N5}$ | Device-specific luma threshold $T_{L,D}$ for LG Nexus 5 (N5). | $lux$ | 1.0, **2.0***, 3.0, 4.0, 5.0 |
| $T_{L,S2}$ | Device-specific luma threshold $T_{L,D}$ for Samsung Galaxy S2 (S2). | $lux$ | 50.0, 55.0, **100.0***, 150.0 |
| $T_{L,V}$ | Video-based algorithm to distinguish low from high brightness video segments. | - | 45, **50***, 55, 60, 65 |
| Camera Shake Assessment | | | |
| $\alpha$ | Low-pass filter cut-off threshold for camera shake detection using the linear accelerometer. | - | 0.1, 0.2, **0.3***, 0.4 |
| $T_{S,V}$ | Video-based algorithm threshold for detection camera shakes. | - | 0.06, 0.10, **0.14***, 0.18, 0.22 |
| $T_{S,AS}$ | Threshold for auxiliary sensor-based camera shake detection algorithms. | $\frac{m}{s^2}$ | 0.1, **0.2***, 0.4, 0.6, 0.8, 1.0 |
| $T_{lowGray}$ | Threshold for detecting low gray intensity values in the video-based camera shake detection. | - | 30 - 126 (in steps of 4) **98*** |
| Harmful Occlusion Assessment | | | |
| $T_{ED}$ | Edge density for a subblock needs to be lower than the reference models edge density minus this threshold. | none | 0.1, 0.3, **0.4***, 0.5 |
| Camera Misalignment Assessment | | | |
| $T_{CM,AS}$ | Degradation score threshold for rotation detection | | 0, 5, 15, **25***, 35, 55, 75 |

Table 8: Parameter study for the proposed camera shake, harmful occlusion, and camera misalignment algorithms, including thresholds for auxiliary sensor-based and video-based algorithms. * depicts the parameter configuration which achieved the highest F1-score.

$T_{L,D}$ and $T_{L,V}$ are used as thresholds for the adaptation of the camera shake algorithm based on both the smartphone's light sensor and video-based processing. $T_{L,N5}$ and $T_{L,S2}$ are device specific thresholds for the Nexus 5 and Samsung S2 respectively.

Different values were tested, as stated in Table 8. It was empirically determined that the low brightness values are $T_{L,N5} = 2.0$ and $T_{L,S2} = 100.0$ and thus very different for different devices. As a fallback solution, $T_{L,V}$, is the luma intensity threshold for pixels which we have empirically determined to be equal to 50. It is used, when no light sensor data is available.

For determining whether a camera shake is present or not the thresholds $T_{S,V}$ and $T_{S,AS}$ are introduced. The threshold $T_{S,V}$ is responsible for classifying imperceivable and perceivable shakes in the video-based algorithm. Thresholds range from $[0, 1]$. From enumerating the parameter values, the highest F1-score could be achieved at a threshold of $T_{S,V} = 0.14$.

13 All videos are from the event NAF_160312.

14 A total of four videos from events *2_VilanovaRambla* and *3_MiniTrain*

For the auxiliary sensor-based camera shake assessment, the linear acceleration values are used. Small values detected by this sensor are due to unintended movements, but do not significantly degrade the perceived quality. The respective threshold $T_{S,AS} = 0.2$ achieves the most precise and reliable detection of camera shakes.

*Parameters for Harmful Occlusion Detection*

The adaptation used for camera shake assessment is also applied to the harmful occlusion algorithm selection ($T_{L,D}$). Whereas the tracking-based harmful occlusion assessment algorithm detects the harmful occlusions without any adjustable parameters, the threshold $T_{ED}$ is relevant for edge density-based occlusion detection. In reference to a previously determined value for the edge density in a subblock of a frame, this threshold gives a safety corridor in which varying edge densities do not indicate a harmful occlusion. The variations studied are detailed in Table 8. The highest F1-score could be achieved for a threshold of 0.4.

*Parameters for Camera Misalignment Detection*

Smaller tilts will not significantly degrade the user experience when watching the video. Thus, the threshold $T_{CM,AS}$ depicts a barrier, which classifies if a tilt is disturbing or not. From the parameter study, a tilt of 25° and above is classified as a camera misalignment which degrades the experience.

#### 4.4.1.3 *Evaluating the Camera Shake Assessment*

The evaluation of the camera shake assessment consists not only of the proposed algorithms, but also of state-of-the-art algorithms for comparison. After the introduction of the algorithms, the performance is assessed regarding the F1-score and runtime.

*Evaluated Algorithms*

The proposed algorithms for camera shake assessment are abbreviated as $CS_V$ for the video-based, $CS_{AS}$ for auxiliary sensor-based, and $CS_{Hybrid}$ for the proposed hybrid camera shake assessment algorithm. In addition, related approaches of Campanella et al. [**Campanella2007**], Saini et al. [**Saini2012**], Bano et al. [**Bano2015**] are used in the comparison.

*F1-Score of the Algorithms*

The results of the evaluation are depicted in Table 9. The table describes different versions of the proposed algorithm using single sensor input ($CS_V$ and $CS_{AS}$) and the hybrid algorithm ($CS_{Hybrid}$). From the three different datasets, it is obvious that the adaptive, hybrid algorithm leads to improved precision as well as recall - thus, generating the highest F1-score. For UGVD, the $CS_{AS}$ already achieves an F1-score of 1.0, which is met by $CS_{Hybrid}$. In all other datasets, the dynamic switching between video-based and auxiliary sensor-based search for camera shakes results in a more reliable detection. At the same time, the algorithms achieve a higher precision and recall compared to the related approaches. The only exception is the video-based algorithm ($CS_V$) and Bano et al.'s approach [**Bano2015**], called $CS_{Bano}$ for the CMDG dataset.

| | *CMDG* | | | | *JIKU* | | | | *UGVD* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithm | P | R | F1 | RTS | P | R | F1 | RTS | P | R | F1 | RTS |
| $CS_{Campa.}$ | 0.11 | 0.82 | 0.193 | 0.905 | 0.92 | 0.93 | 0.924 | 0.536 | 0.03 | 0.777 | 0.057 | 0.3131 |
| $CS_{Saini}$ | 0.1 | 0.7 | 0.175 | 0.904 | 0.92 | 0.81 | 0.861 | 0.534 | 0.29 | 0.77 | 0.055 | 0.31 |
| $CS_{Bano}$ | 0.41 | 0.57 | 0.477 | 0.0002 | ○ | ○ | ○ | ○ | 0.117 | 0.285 | 0.166 | 0.0002 |
| $CS_{V}$ | 0.2631 | 0.3061 | 0.282 | 0.96 | 0.923 | 0.969 | 0.946 | 0.6363 | 0.59 | 0.601 | 0.595 | 0.31 |
| $CS_{AS}$ | 0.53 | 0.73 | 0.616 | 0.0001 | 0.93 | 0.8378 | 0.881 | 0.0002 | 1 | 1 | 1 | 0.0001 |
| $CS_{Hybrid}$ | 0.53 | 0.73 | 0.614 | 0.48 | 0.92 | 0.99 | 0.9537 | 0.317 | 1.0 | 1.0 | 1.0 | 0.156 |

Table 9: Camera shake assessment results regarding precision (P), recall (R), F1-score (F1) and the runtime regarding the quota of duration of the assessment in relation to the duration of the dataset. This is called real-time suitability (RTS), where a value below 1 indicates a real-time calculation.

For the CMDG dataset, it can be observed that the higher F1-scores of the proposed algorithms $CS_{AS}$ and $CS_{Hybrid}$ are achieved by increased precision, while keeping a similar recall rate as the related work ($CS_{Saini}$). For the JIKU dataset, the precision of all algorithms is at a similar level: between 0.92 and 0.93. The recall rates of the proposed hybrid algorithm are close to 1 when combining the video-based and auxiliary sensor-based algorithms, reducing false hits. Consequently, the F1-score is close to 1. Here, the gyroscope-based algorithm $CS_{Bano}$ outperforms the proposed video-based algorithm, as the dataset was constructed during the development of CMDG. The scenes recorded for this dataset suffer from bad lighting conditions and rapid, non-camera shake motion. The gyroscope recordings have been perfectly synchronized with the video. This brings us to the conclusion that such an algorithm will perform significantly worse in imperfect scenarios. As a result, the performance of $CS_{Bano}$ degrades in comparison to the proposed algorithms for the UGVD dataset. The devices that recorded the JIKU dataset did not offer gyroscope sensings.

The remaining and competing video-based algorithms lack precision, and show a higher runtime in comparison to the proposed algorithm. In conclusion, the hybrid algorithm offers the highest potential to reliably detect camera shakes, as it can rely on a fast and precise auxiliary sensor-based algorithm; when lacking such input, the hybrid algorithm can apply a sophisticated video-based algorithm instead.

*Runtime of the Algorithms*

The runtime of the algorithms is depicted in Table 9. It shows that the purely auxiliary sensor-based algorithms $CS_{Bano}$ and $CS_{AS}$ show a high correlation and achieve a low runtime. Here, the approach by Bano et al. **[Bano2015]** outperforms all proposals, but if not all recording devices (e.g., older smartphones) have a gyroscope no calculation is possible. The hybrid algorithm builds a good trade-off, as it offers the ability to achieve an RTS (real-time suitability) between 0.3124 and 0.65, where a lower value is more favorable.

#### 4.4.1.4 *Evaluating the Harmful Occlusion Assessment*

Harmful occlusion detection and assessment are introduced in Section 4.2.3 and is evaluated in comparison to related video-based algorithms.

*Evaluated Algorithms*

As discussed in the related work chapter (Chapter 2), some harmful occlusion detection algorithms exist, especially in the area of object tracking. No existing algorithm quantifies the impact on the subjective perception of a video. The proposed hybrid algorithm, which is discussed in Section 4.2.3, is termed $HO_{Hybrid}$. The approach by Saini et al. [**Saini2012**] is termed $HO_{Saini}$. This algorithm analyzes each video frame in an edge representation and classifies parts of the video frame as occluded if the edge density decreases.

Also, the approach of Saravanakumar et al. [**Saravanakumar2012**] is used for comparison ($HO_{Sarav}$). As discussed above, they apply a continuous contour-tracking-based approach for the detection of harmful occlusions. It removes shadows and background information to reliably detect and track objects.

*F1-Score of the Algorithms*

| | *CMDG* | | | | *JIKU* | | | | *UGVD* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithm | P | R | F1 | RTS | P | R | F1 | RTS | P | R | F1 | RTS |
| $HO_{Saini}$ | 0.471 | 0.053 | 0.096 | 11.25 | 0.614 | 1 | 0.76 | 1.324 | 0.521 | 0.335 | 0.408 | 0.382 |
| $HO_{Sarav}$ | 0.222 | 0.145 | 0.175 | 1.012 | 0.121 | 0.6787 | 0.205 | 0.2981 | 1 | 0.1787 | 0.303 | 0.433 |
| $HO_{Hybrid}$ | 0.88 | 0.988 | 0.931 | 0.942 | 0.615 | 1.0 | 0.762 | 0.6644 | 0.74 | 0.6 | 0.662 | 0.347 |

Table 10: Harmful occlusion assessment results regarding precision (P), recall (R), F1-score (F1), and the runtime regarding the quota of duration of the assessment in relation to the duration of the dataset called real-time suitability (RTS), where a value below 1 indicates a real-time calculation.

In contrast to the other degradations, the harmful occlusion assessment solely relies on video as input. Related approaches regarding harmful occlusion detection act similarly to the proposed approach, analyzing either object movements or the loss of edge density. The proposed approach dynamically adapts between different video-based algorithms and increases their robustness against brightness changes and badly illuminated videos, which are common in today's UGV datasets.

For the CMDG dataset, the proposed hybrid algorithm outperforms the related approaches, both regarding precision and recall rate. In this context, the proposed algorithm handles such situations better. It outperforms all other approaches in the CMDG dataset. These advantages decrease when the JIKU dataset or the UGVD dataset is considered. For the JIKU dataset, there is an advantage compared to the algorithm $HO_{Saini}$, both regarding precision and recall. For the novel UGVD dataset, the tracking-based algorithm $HO_{Sarav}$ outperforms the other algorithms regarding precision. The reason is that in comparison to the other datasets, no camera motion such as panning or tilting disturbs the object tracking. The algorithm wrongly classifies contour losses as harmful occlusions.

In the evaluated datasets, the proposed hybrid algorithm achieves F1-scores between 0.66 and 0.762 and still outperforms Saini et al.'s approach in their JIKU dataset. Especially, the recall rates in the JIKU dataset reach up to 1.0.

*Runtime of the Algorithms*

The tracking-based approach, as proposed in $HO_{Sarav}$ or in $HO_{Hybrid}$, require higher processing times in comparison with edge density-based approaches. The proposed hybrid algorithm extends from the ideas of $HO_{Sarav}$ and $HO_{Saini}$. It improves precision

and omits unnecessary steps which reduces the runtime. It favors rapid processing of the quality assessment. As a result, the hybrid algorithm always achieves real-time processing of the content. In the presence of 480p and 1080p content as in the CMDG dataset, it is the only algorithm that falls below the RTS of 1. For the remaining, low-resolution datasets, processing time is lower. In general, one could say that the differences between the hybrid algorithm $HO_{Hybrid}$ and $HO_{Sarav}$ are nearly imperceivable in all cases - but the precision of the proposed algorithm is higher.

#### 4.4.1.5 *Evaluating the Camera Misalignment Assessment*

Camera misalignment is classified into the quick detection of camera tilts and the detection of a collaboratively sensed RoI captured in a scene. Related algorithms are introduced and compared with the proposed algorithms.

*Evaluated Algorithms*

For camera misalignment assessment, a video-based $CM_V$, auxiliary sensor-based $CM_{AS}$, and hybrid algorithm $CM_H$ are proposed. The tilt detection algorithm integrated into the Android OS is evaluated, too. It leverages accelerometer values and determines a tilt if one of the axes of the accelerometer indicates a turn of at least 25°. This algorithm is termed $CM_{AD}$.

Furthermore, the algorithm for tilt detection introduced by Saini et al. [**Saini2012**] is evaluated. It analyzes video frames for their horizontal alignment, and acts similar to our approach analyzing the edge orientation ($CM_{Saini}$).

*F1-Score of the Algorithms*

| | *CMDG* | | | | *JIKU* | | | | *UGVD* | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Algorithm | P | R | F1 | RTS | P | R | F1 | RTS | P | R | F1 | RTS |
| $CM_{AD}$ | 0.12 | 0.35 | 0.178 | 0.00002 | 0.0538 | 0.7783 | 0.1 | 0.00002 | 0.016 | 0.177 | 0.029 | 0.00002 |
| $CM_{Saini}$ | ○ | ○ | ○ | 0.9525 | 0.584 | 0.103 | 0.176 | 0.561 | 0.2395 | 0.469 | 0.317 | 0.26 |
| $CM_{AS}$ | 1.0 | 1.0 | 1.0 | 0.0002 | 0.949 | 0.858 | 0.901 | 0.0001 | 0.865 | 0.866 | 0.865 | 0.0001 |
| $CM_{Hybrid}$ | 1.0 | 1.0 | 1.0 | 0.952 | 0.945 | 0.86 | 0.9 | 0.561 | 0.875 | 0.933 | 0.903 | 0.2584 |

Table 11: Camera misalignment assessment results regarding precision (P), recall (R), F1-score (F1), and the runtime regarding the quota of duration of the assessment in relation to the duration of the dataset. This is termed real-time suitability (RTS), where a value below 1 indicates real-time calculation.

The proposed camera misalignment algorithms are based on either auxiliary sensor readings ($CM_{AS}$), specifically the linear accelerometer or a hybrid combination with the video-based algorithm $CM_{Hybrid}$. $CM_{Saini}$ achieves no valid results for the CMDG, as low brightness conditions seem to affect the performance of the algorithm.

For the remaining algorithms and datasets, none of the related approaches achieves similar detection rates of camera misalignments and tilts. Except for the JIKU dataset where the recall rate of $CM_{AD}$ reaches 0.7783, the metrics for the related approaches are always below 0.5. The two related approaches thus fail when being run on the respective datasets. Especially, for the auxiliary sensor-based algorithm, small changes could significantly improve the results from the algorithm $CM_{AD}$ to $CM_{AS}$. An auxiliary sensor-based approach $CM_{AS}$ already achieves a quite reliable detection of tilts. The hybrid algorithm $CM_{Hybrid}$

improves the results and increases the runtime only slightly. In all cases the hybrid as well the proposed auxiliary sensor-based algorithm achieve F1-scores of 0.86 and higher. In many cases, the increased runtime of the $CM_{\texttt{Hybrid}}$ may not be worth the only minimal improvements in comparison to $CM_{AS}$.

*Runtime of the Algorithms*

Considering the runtime of the algorithms: all algorithms can be executed in real-time. The auxiliary sensor-based algorithms are again executed most quickly, but the hybrid algorithm $CM_{\texttt{Hybrid}}$ achieves a similar performance with an RTS between 0.2584 and 0.945. The variance in the runtime can be explained by the varying dataset resolutions and needs for applying the video-based part in the $CM_{\texttt{Hybrid}}$ algorithm.

### 4.4.2 *Assessing the Performance of the PaSC*

#### 4.4.2.1 *Setup of the Evaluation*

In comparison to the assessment of the precision and runtime of the quality assessment algorithms, this evaluation is conducted in real-world, distributed deployment with multiple devices. The potential for scalability of a centralized, concurrent processing of quality assessment algorithms in comparison to a distributed execution on mobile devices and central servers is assessed. An application is designed, which consecutively requests quality assessments from the PaSC with timing deadlines. The goal is to increase the rate of completed quality assessments that delivered their results on time.

*Algorithms*

To assess the performance of the PaSC, we not only use the discussed Recording Quality Assessment (RQ), but also available algorithms for Audio Quality Assessment (AQ) and Video Quality Assessment (VQ). Each category of algorithms is evaluated independent of the others, offering all resources of our setup solely to the respective assessment category.

The recording quality assessment algorithms address the degradations of camera shake, camera misalignment, and harmful occlusions. Different algorithms were proposed to analyze and measure the effects of compression in digital video. Open-source implementations of NR algorithms such as BRISQUE [**Mittal2011**] (BR), V-BLIINDS (VB) [**Saad2014**], Shrestha et al. (SHR) [**Shrestha2010**], Campanella et al. (CA) [**Campanella2007**], and Saini et al. (SAI) [**Saini2012**] are used.

The analysis of the audio track of UGV is discussed. Besides video quality assessment, algorithms were designed for quality assessment of the audio tracks [**ITU-P863**], speech quality analysis [**ITU-P563**], or specifically, audio degradations in UGV [**Li2013**].

*Device setup*

The setup of our evaluation consists of a server (Ubuntu 14.04) as well as 15 mobile devices consisting of one Samsung Galaxy S6 (Android 5.1.1), one Sony Xperia Z3 (Android 5.1.1), eight Nexus 5 (Android 6.0) and five Nexus 4 devices (Android 5.1.1). The devices are categorized into high-end servers and low-end, medium and high-end smartphones. The

classification was validated by the results of the Vellamo[15] and AnTuTu benchmarks[16] (see Table 12).

Table 12: PaSC Evaluation: Performance benchmark results for Vellamo and AnTuTu and categorization of the devices.

| | Vellamo | AnTuTu | Class | No |
|---|---|---|---|---|
| LG Nexus 4 (N4) | 600 | 16749 | low-end | 5 |
| LG Nexus 5 (N5) | 1166 | 26340 | medium | 8 |
| Sony Xperia (Z3) | 1551 | 43911 | high-end | 1 |
| Samsung Galaxy S6 edge (S6) | 1569 | 68830 | high-end | 1 |
| Server | Intel Xeon (6 cores) 64 GB memory | | server | 1 |

*Network Setup*

The evaluation setup ensures that the server hosting the algorithm repository is located in the same geographic region - but remote from the mobile devices. All runtime statistics are collected on the server, too.

Traffic shaping is used to set the maximum data rate to 50 $\frac{\mathrm{MBit}}{\mathrm{s}}$ and a randomly selected minimum latency between 100-300 milliseconds per transmission **[Lampe2013]**. Higher latencies, e.g., as invoked by the access or backbone network, are included in our results. To achieve reliable results, the evaluations were re-performed ten times each. Furthermore, evaluations were scripted with the same configuration parameters, under similar conditions using the same videos.

*Scenario*

The evaluation describes an experimental setup in which quality calculation requests are sent continuously by the devices to trigger the selection of an optimal combination of algorithm and device. At any time 15 assessment requests are processed - thus one request per device. Once completed, another request is created. The deadlines and accuracy requirements are normalized in the range $[0, 1]$. 0 represents the lowest and 1 the highest processing time. The statistics are offered by the algorithm repository of the PaSC. PaSC is compared in both evaluations with an assessment of all algorithms on a central server.

Furthermore, assessments of different quality assessment categories (AT) can be run in parallel. For each of the evaluated cases metrics are determined - including the average runtime of the quality assessment, the number of successful runs in a given time, and the average utilization. For the assessment of the runtime, the total time from the request for a quality assessment until the delivery of the result (including transmission and algorithm execution times) is given. Successful runs are defined as algorithm calculations processed in time for a given deadline. The number of successful runs and its relation to the completed runs are metrics indicating if system load and application requirements allow a timely completion of the assessment. The utilization metric indicates how many resources of the individual devices are used in average.

15 Qualcomm Vellamo Metal Benchmark; Visited on: 10/09/2016

16 AnTuTu Benchmark; http://www.antutu.com/en/index.shtml, Visited on: 10/09/2016

#### 4.4.2.2 *Utilization of the Devices*

The results of this evaluation are depicted in Figure 29. In a first step, the increased utilization by an intelligent selection and placement of the algorithms is shown. For comparison, a central server instance as described in the system setup section is used. As quality assessment requests are made by the application, the average utilization of this single server setup is close to 100% (97.1%). The rationale behind this is quite obvious: As the number of devices requesting quality assessments is higher than the number of available processors, the resources of the single server are exhausted. For the calculation of total system

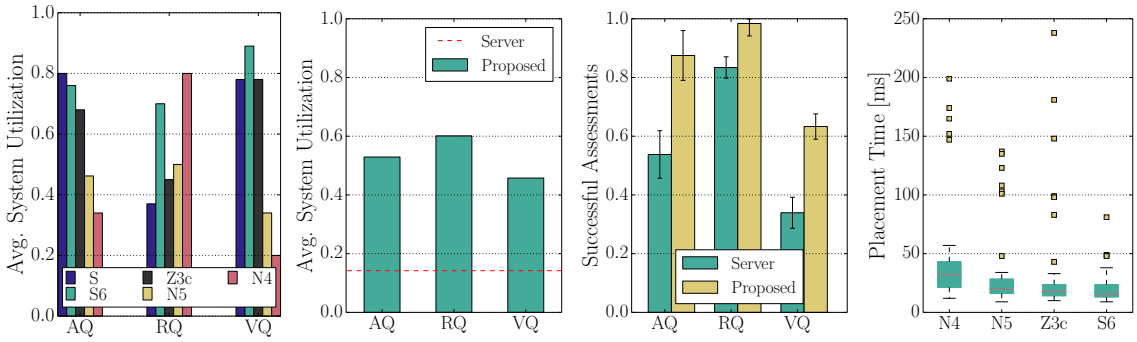



Figure 29: (a) Average utilization of the different devices when using the PaSC - S: Samsung Galaxy S6 edge, N4: LG Nexus 4, N5: LG Nexus 5, Z3: Sony Experia Z3, S: Server; (b) System utilization averaged and normalized across all devices in the evaluation setup in comparison with a single server; (c) Number of successful quality assessments when hard-timed deadlines are given; (d) Calculation overhead in [ms] for making and distributing a decision in PaSC of which algorithm to run on which device.

resources, the CPU utilization of all devices is sampled with Network Time Protocol (NTP) synchronized clocks. The different processor features (architecture, clock rate, the number of cores) were normalized to retrieve a total utilization score.

In comparison to the single server calculation, Figure 29 illustrates that the PaSC distributes different quality assessment tasks to the mobile devices and thus causes a high utilization of resources. Figure 29 (a) illustrates the average utilization per device and assessment category when using PaSC. The most obvious observation that can be drawn from this figure is that quality assessment tasks are assigned to the devices independent of specific device capabilities. None of the devices reaches its utilization limit (1.0). Especially, weaker devices such as the N4 have only a limited amount of quality assessment tasks in the categories VQ and AQ, as all algorithms rely on resource-intensive video- or audio-based algorithms. The PaSC is given timing deadlines, which a device must comply with. Thus, weaker devices are rather not selected if the algorithm would require a significant processing time.

As the proposed algorithms for RQ are using auxiliary sensor input and the related algorithms are computationally less expensive, the quota of assessments on N4 is significantly higher. At the same time, average utilization of the remaining devices drops. For RQ, it is obvious that offloading works well as the average utilization of the server drops below 0.4. One of the reasons for the utilization for RQ algorithms is that many requests can be executed locally without distribution to remote devices.

Figure 29 (b) depicts, as a result, an average system utilization normalized for all devices. It depicts the utilization of a single server as a red, dotted line in comparison to the av-

erage utilization when using the PaSC. As the number of highly accurate yet lightweight algorithms increases, total system utilization increases significantly. The total utilization increases to around 60% for RQ in comparison to a single server installation where only 14.4% of the system resources are leveraged. Even for the VQ algorithms, the total system utilization increases to above 40% - thus, more than doubled. In all cases, utilization increases significantly and the high number of N5 and N4 devices boosts the average utilization for RQ in comparison to AQ and VQ. Still, one can conclude for the utilization that for all assessment categories, PaSC finds sufficient assessment requests to be run on the mobile devices, which significantly reduces the load on the server and enhances scalability.

#### 4.4.2.3 *Increased Completion of Quality Assessments*

A measure for the effectiveness of the PaSC is the number of completed quality assessments. PaSC is perceived as beneficial, if the number of completed quality assessments is significantly higher compared to the centralized processing. Figure 29 (c) illustrates a comparison of the single server and the proposed solution regarding successful quality assessments. Following the description in Section 4.4.2.1, a successful quality assessment represents a complete execution of an algorithm and the reporting of the quality result to the requesting application.

An obvious result is that the ratio of successful (in-time, processed) assessments increase significantly. In particular, the complex quality assessment categories VQ and AQ benefit from the distributed calculation, increasing their rate of successful assessment by 0.2937 and 0.3371. Even though they can place only a small number of runs on the weak devices (especially the N4), the more powerful devices such as the S6 and Z3c help to increase the total number of completed assignments. For the computationally inexpensive algorithms of the RQ category, the ratio increased by 0.1498.

This offloading allows to process more quality algorithms in time. Note that to allow a fair comparison for evaluating the single server and PaSC the total number of assessments is kept constant between the two scenarios. The PaSC would have achieved a significantly higher number of completed assessments in a given time.

#### 4.4.2.4 *Influence of Device Heterogeneity*

Besides the effectiveness of the PaSC, its efficiency shall be evaluated by inspecting the costs of running the PaSC in terms of processing delay and caused network traffic. Figure 29 (d) shows system processing times, which consist of processing statistics to select an algorithm and the time needed to decide on the algorithm placement. The time of informing a device to start the processing is also included.

Interestingly, the deployment to different devices affects this decision. Especially, the low-end devices (N4) need considerably longer time (with a mean of 40 milliseconds) than the high-end devices (S6, Z3c). Yet, all devices show a significant proportion of outliers which result in selection times of up to 250 milliseconds. Such high processing times significantly affect the quality assessments if they are needed for real-time assessments. In most cases, the selection and placement processing times take less than 50 milliseconds, which is negligible in comparison to the processing times of the algorithms. For example, the delay invoked by transmitting sufficient video for algorithm execution lies around 137.5 milliseconds ($\sigma^2 = 27.2$) per video. The coordination data overhead necessary for using the PaSC is rather small at around 20.18 kB to 20.41 kB per request. If we consider an average bit rate of $1500\frac{\mathrm{KBit}}{\mathrm{s}}$ for a video stream, the coordination overhead is considerably lower than the video data.

## 4.5 CONCLUSION

An essential step for many multimedia applications is assessing the quality of video streams. This chapter discusses a scalable and adaptive quality assessment module. It can run arbitrary quality assessment algorithms on mobile devices or servers in a way, that multimedia applications can set requirements for the execution. The PaSC is a module that selects from a set of algorithms the best one according to specified requirements. The selected algorithm is placed on a processing device which ensures in-time completion and fulfillment of application requirements. PaSC offers a scalable solution to achieve the selection of the best algorithm and device at a given time to perform a quality assessment. While inspecting the different quality assessment algorithms, a significant lack of algorithms for the detection and assessment of degradations was identified. PaSC offers a set of algorithms to detect and assess the impact of the degradations: camera shaking, camera misalignment, and harmful occlusions. A set of algorithms was proposed, which rely on either a single sensor input (i.e., a camera) or a combination with other auxiliary sensors, such as an accelerometer or a gyroscope. The proposed algorithms show superior performance and reduced runtime - and they are the first to quantify the decrease in quality of recording degradations.

# 5 MOBILE VIDEO UPLOAD

This chapter describes a novel MBS that offers content-adaptive uploading of media streams. Many existing MBSs rely on the unadaptive video transmission using RTMP, which results in the need for consistently high throughput rates. The proposed Live Video Upload System (LiViU) uses adaptive video streaming to deal with changing network conditions. In addition, LiViU reacts to changing application requirements and different scenarios by switching mechanisms. The term mechanism is derived from the Future Internet project MAKI[1], and specifies single or multiple protocols offering a specific function within a communication network [**Gross2013**]. In this thesis, the replacement of a running mechanism with another mechanism offering a similar functionality is called mechanism adaptation [**Froemmgen2015**][2]. Mechanism adaptations are necessary, as LiViU has not only to deal with varying network conditions but also heterogeneous streaming scenarios (i.e., remote, in situ and hybrid streaming), and support for different applications. The various applications discussed in this thesis are the PaSC, which leverages in situ and remote devices for efficient quality assessment of media streams (see Chapter 4), and the video composition proposed in Chapter 6. By combining the content and mechanism adaptation, LiViU better copes with unsteady, resource-capped networks and changing application requirements.

Mechanisms used in LiViU are selected based on a study of different upload protocols. To select the most beneficial mechanisms, simulative studies with state-of-the-art MBSs are performed. It is shown that existing mechanisms have different strengths as well as weaknesses depending on the environmental conditions. The strengths of different mechanisms influence the design of LiViU, that can dynamically adapt between mechanisms to not only allow a high bit rate, but also a low delay in streaming. A prototype on the Android OS is proposed and evaluated, which combines mechanism and content adaptation.

Concepts and ideas discussed in this chapter revise the peer-reviewed publications [**Stohr2016**, **Wilk2014c**, **Wilk2016f**, **Wilk2015d**].

## 5.1 MBS SYSTEM MODEL

The MBS scenarios discussed in Section 2.4.2 are classified into remote, in situ, and hybrid streaming. What the scenarios have in common is that they consist of at least one recording device transmitting video to a receiver, which can be either a smart mobile device or a remote server.

### 5.1.1 *Recording Device*

The recording device consists of two important components: the video recording API and the recording buffer.

1 Deutsche Forschungsgemeinschaft Collaborative Research Cluster 1053 on Multi-mechanism Adaptation in the Future Internet (MAKI)

2 According to the terminology of the Future Internet project MAKI, the mechanism adaptations described in this chapter can also be termed as a transition between mechanisms.

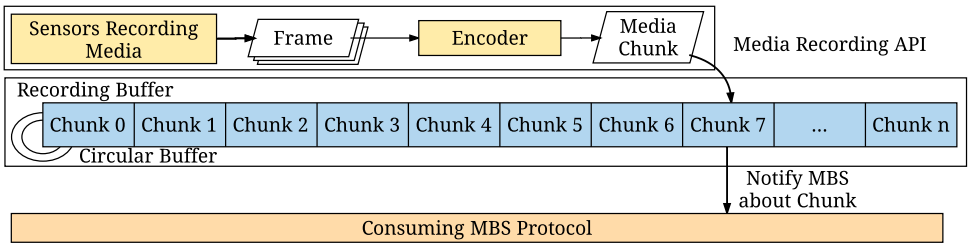



Figure 30: Example of a Media Recording API and Recording Buffer.

#### 5.1.1.1 *Media Recording API*

Each MBS has access to media capturing sensors. It is assumed that these sensors map potentially analog signals into digital data. This step requires the support of a video codec.

A digital video stream is made available in an encoding step that ensures a suitable compression of the data, so that it is feasible to be streamed in today's mobile networks. Typical encodings supported by smart mobile devices are H.264/AVC **[Wiegand2003]** or H.265/HEVC **[Sullivan2012]**. An additional requirement is that available smartphones have the capability to encode and decode video in real-time. This is possible if the devices posses a video codec circuit, which enables en- and decoding in hardware. Today, such a hardware support is available for many devices when using H.264/AVC **[Wiegand2003]**. A final requirement describes that a video codec supports streaming - meaning that parts of a video can already be played without possessing the complete video. This is essential for live streaming, but not supported by many video containers. The non-encapsulated, raw H.264/AVC and H.265/HEVC, as well as the video container MPEG transport stream (MPEG-TS), support this requirement. As a result, the media recording API provides a streamable, compressed video stream to the recording buffer.

#### 5.1.1.2 *Recording Buffer*

The recording buffer is the interface to the MBS. It is constantly filled with video from the media recording API (see Figure 30) and read by the MBS. A circular buffer is used, which processes data in First-In First-Out (FIFO) order and does not consume more memory than the smart mobile device can offer. The size of the video recording buffer is limited to $w_{SB}$ video chunks. A filled buffer is overwritten in FIFO order, resulting in the loss of to-be-transmitted video chunks. Thus, each available chunk in the recording buffer must represent an independently decodable video segment.

The media recording API consecutively writes video chunks to the recording buffer, e.g., when recording a video frame. As soon as a decodable chunk is available, the recording buffer informs the uploading mechanisms in the MBS to transfer the video chunk. In most cases, this video chunk is passed instantly, but stored in the recording buffer in order to allow retransmission if a message is lost. Thus, the buffer supports random access, which allows for up to the last $w_{SB}$ chunks to be retransmitted. If the upload rate is below the bit rate of the recorded video, data can be consumed by the MBS as soon as all preceding chunks are processed.

### 5.1.2 *Receiver*

Receivers of the video stream have a buffer for storing incoming video chunks. It builds the abstraction to the higher layer application sink, which processes video chunks.

#### 5.1.2.1 *Receiver Buffer*

Similar to the recording device, a buffer is assumed on the receiver side. It stores video chunks until they are consumed by a multimedia application. The receiver has a limited window of cached video chunks of size $w_{RB}$. The processing is similar to the recording side in FIFO order and uses a circular buffer. In contrast to the video sending device, the receiver manages multiple buffers in relation to the number of actively streaming senders. All receiving buffers are available to the application sink and have to be kept in sync.

#### 5.1.2.2 *Application Sink*

An application sink accesses a video buffer, and further processes the video. On the receiver side, applications consuming the video stream can be video players, video streaming servers, or sophisticated multimedia applications such as video composition systems. Video players consume the video stream instantly by decoding and visualizing it on a display. This is a common approach for the in situ streaming scenario as discussed in Section 2.4.4. The video stream can also be received by a video streaming server, which ingests it into a distribution network for remote receiving and playback. As an example for a multimedia application, a video composition system is discussed in Chapter 6 and in the previous chapter describing the PaSC.

### 5.1.3 *Further Assumptions for the System Model*

It is important to know that modeling of lower layer protocols is not part of this thesis - as it focuses on the MBS design on the application layer of the ISO/OSI model. Furthermore, standard protocols are used on the transport layer, i.e., TCP and UDP. No modifications on the lower layers are required. It is assumed that the functionality is managed by an OS and can be accessed by standard Berkeley sockets **[ISOIECIEEEStandard99452009]**.

## 5.2 STUDY ON ADAPTATIONS IN MBSES

In a study of the advantages and disadvantages of different MBSs, mechanisms and concepts are identified which should be supported by LiViU. The study is performed in a simulation environment, allowing for a large-scale analysis of the potential for adaptation.

### 5.2.1 *Video Upload Protocols*

Five upload protocols, including industry standards and research proposals, are investigated for their use in an adaptive manner. The focus lies on the leveraged transport mechanisms, scheduling of video chunks, and coordination of streaming session participants. Aspects such as security, encryption, and Digital Rights Management (DRM) are out of the scope of this thesis.

#### 5.2.1.1 *Real-Time Messaging Protocol (RTMP)*

The first protocol investigated is the de-facto standard for MBS, as they are used by the major platforms YouNow, Twitch.tv, Periscope, YouTube.Live and Facebook.Live. A detailed discussion of RTMP is given in Section 2.4.6.2. For the classification of the protocol, it is important to know that it uses TCP as the transport layer protocol and relies on a push-based delivery of media chunks from the recording device.

#### 5.2.1.2 *Real-Time Media Flow Protocol (RTMFP)*

Derived from RTMP, the RTMFP [**rfc7016**] replaces the connection-oriented TCP transport mechanism with the datagram-oriented UDP. Video streams are transmitted on top of a connection in flows that are pushed from a recording device to a receiver. Similar to RTMP, these media flows are message-based. It provides a higher speed for connection restoring and IP mobility support which is not supported by the connection-oriented TCP. In contrast to the RTMP design, RTMFP has been designed for client-server as well as for P2P systems. Underlying the unidirectional flows are bi-directional connections, which are established in an initial handshake procedure.

RTMFP is session-based and establishes the session in two round trips [**rfc7016**]. A sender initiates the session by sending a single *IHello* (”Initiator Hello”) message. The response contains a cookie, which allows the second round trip and final session establishment, known as keying. Another send-and-response round establishes an optionally encrypted video session. After the response is received, the initial video chunks can be exchanged. The behavior for session establishment is quite similar to the procedure of RTMP, but RTMFP has to handle message losses in the application layer.

To ensure a reliable transmission over UDP, RTMFP assumes that the receiver acknowledges all messages. RTMFP implements a congestion control that needs to comply with Internet recommendations [**rfc2914**] and which is not allowed to be more aggressive than TCP’s slow start algorithm. The protocol requires application layer solutions for coping with packet losses and an in-order transmission of video chunks. Due to the reduced overhead and latency of UDP-based protocols, we assume that it is better suited for the live streaming applied in MBS.

#### 5.2.1.3 *DASH Upload (DASH-U)*

In recent years, HTTP-based video streaming approaches have gained significant interest. The MPEG DASH [**Stockhammer2011**] standard defines network communication using HTTP (TCP) as well as the description of the video in a manifest, called the MPD. This pull-based video streaming scheme is used for the design of DASH-U. Using DASH in an MBS requires that the receiver requests video segments. These requests can regularly be planned, e.g., with a fixed frequency, or they can be event-based, e.g., after a DASH segment is received. Also, the video stream receiver can specify rules to request only specific video segments. Each client transmits segments of a video only if they are requested by the receiver. The underlying request-response (pull-based) communication pattern is very costly in terms of overhead. On the other hand, the method is well-suited for scenarios where the receiver requires only a few video segments from each source. The control of what video stream is requested at which point in time lies at the server.

#### 5.2.1.4 *HTTP POST-based DASH Upload (DASH-P)*

Seo et al. [**Seo2012**] propose the DASH-P as a prototypically evaluated upload protocol. As the name implies, communication in DASH-P is based on the transmission of video segments via HTTP POST requests, i.e., the device pushes video segments. Whereas in DASH-U the server requests individual segments using HTTP GET requests, DASH-P continuously pushes the video segments to the receiver. The overall delay until a segment is available on the streaming server is lower when the two approaches are compared. An advantage of using HTTP instead of a dedicated streaming protocol such as RTMP is that no session or state has to be established before a new segment is transmitted.

#### 5.2.1.5 *UDP-Pull (UDP-PL)*

The majority of the existing algorithms rely on TCP as a transport mechanism. UDP has certain advantages when it comes to live video streaming, as the overhead and latency is lower in comparison to TCP. Pull- and UDP-based streaming protocols are neither widely used nor standardized. We propose a minimal video upload protocol which allows a client to inform a server about an available video stream. The server immediately starts pulling available video segments over UDP. Also, only minimal modifications are made to compensate for the weaknesses of UDP. To allow for the compensation of transmission errors, an Automatic Repeat reQuest (ARQ) concealment technique is implemented (see Section 5.3.3.3). In comparison to RTMFP, this protocol operates with pull-based scheduling and leverages only a minimal session management, i.e., no congestion control, and no encryption or state management.

#### 5.2.1.6 *"Adaptive"*

Also, a protocol is implemented which encapsulates the protocols RTMP, RTMFP, DASH-U, UDP-PL, and DASH-P to always select the best protocol for a given performance metric. The performance metrics implemented and evaluated are described in Section 5.2.2.1. An application can specify the performance metric and triggers the evaluation runs to determine the performance of each protocol. If multiple metrics are defined and evaluated, this allows an application to switch the used performance metric at runtime. The adaptive usage of the protocols is called "Adaptive".

*Extended System Model*

Derived from the system model described in Section 5.1, a component for coordinating and executing the adaptation between the different MBSs is required. On both the sender and the receiver side, buffers act as a decoupling feature from the application, i.e., the video recording API or application sink consuming the video stream. Buffered video chunks are accessed by the transmission layer which is an artificially introduced layer for all mechanisms of an MBS. The transmission layer allows the adaptation of scheduling schemes - either push- or pull-based streaming. In this context, the transmission layer is only aware of the currently used transport protocol, which can either be UDP or TCP. The concept of the transmission layer for the adaptive MBS is depicted in Figure 31.

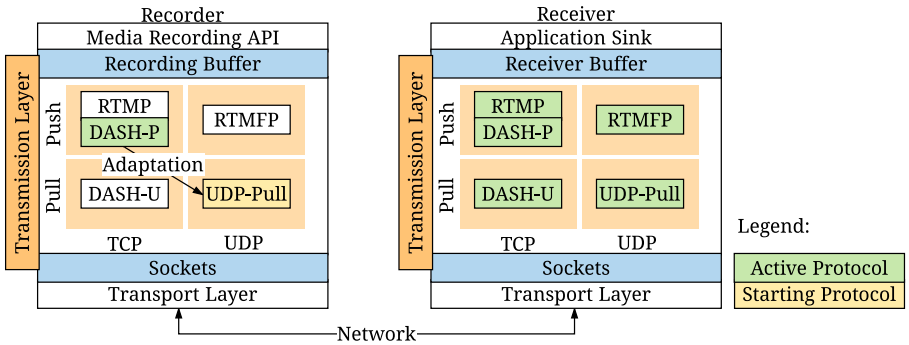



Figure 31: Concept of the transmission layer including the adaptation of MBS upload protocols.

*Contact Management*

Different upload protocols operate within the transmission layer. They have in common that they have to manage their *contacts*. To establish a transmission, the remote addresses are managed as so-called contacts. A contact is a remote device that participates in a streaming session, e.g., by receiving the stream. Contacts represent a unique combination of an IP address and the port of the upload protocol. When run in parallel, different protocols are distinguished by the used network port address. Each protocol has its own, often standardized, port for communication.

*Adapting between the Protocols*

"Adaptive" is configured to switch to a protocol which performs best for a given performance metric (Section 5.2.2.1). A requirement is expressed as a performance metric, which can be measured during the runtime of a protocol. Based on the result of a performance metric, "Adaptive" decides which protocol to choose. The measurement of the metrics and configuration of "Adaptive" to pursue differing requirements at different points in time is discussed in the simulative evaluation in Section 5.2.2.1.

*Adaptive Video Streaming*

"Adaptive" includes the creation of multiple video representations with differing bit rates and their transmission to a receiver. The concept of adaptive video streaming as a form of content adaption has been discussed in Section 2.2.2.1. For the discussion of MBS, it is assumed that the recording device encodes $n$ video representations differing in the bit rate. Each representation is encoded by an SVLC, such as H.264/AVC or H.265/HEVC. At any given moment, only one representation is chosen for transmission. The selection is based on the measured application layer throughput rate for a device, so that the bit rate of a video must be below the available throughput rate.

### 5.2.2 *Assessing the Potential of Transitions between Upload Protocols*

A simulative analysis of the proposed system model is conducted to understand which mechanisms of the upload protocols show superior performance under varying environmental conditions. The findings derived from this simulative study are used for the design of a new MBS in Section 5.3. For simulation purposes, the scenario of remote streaming is chosen.

#### 5.2.2.1 *Performance Metrics*

The used performance metrics of an upload protocol are the overhead of a protocol, the goodput of the protocol, the join time, and the latency for each recorded video segment.

The *overhead* (O) of an upload protocol is calculated in [bits] and is affected by the number and size of the control messages of a protocol as well as the headers of video messages. The average overhead per time unit is used in the MBS, which uses the unit $[\frac{\text{bits}}{\text{s}}]$ and is calculated as $\bar{O} = \frac{O}{t_S}$, where $t_S$ represents the total session time.

The *goodput* (GP) is measured in terms of the effective application layer throughput of video data in $[\frac{\text{bits}}{\text{s}}]$. For a single media track transmission the goodput represents, the average bit rate of the video stream. This excludes protocol overhead or coordination messages. Duplicate video messages are not considered in the calculation of the goodput.

Table 13: Parameters used in the simulative evaluation of different MBSs.

| | |
|---|---|
| **"Adaptive"** | |
| Video Representations: | 500, 750 and 1000 kbit/s |
| Video Adaptation: | every second |
| **Scenario: Concurrent Video Upload** | |
| Parallel recorders: | up to 1000 (trace-based) |
| Number of recorders per Region: | up to 200 (randomly assigned) |
| Number of Regions: | up to 10 |
| Networks: | LTE |
| Upload Bandwidth: | max. 50 MBit/s |
| Latency: | $300 \pm 200$ ms |

To depict the fraction of redundant messages, the duplicate chunk quota DC is calculated as $DC = \frac{N_D}{N_{VM}}$, where $N_D$ is the count of chunks received multiple times, and $N_{VM}$ the number of video messages sent in a complete session.

The *join time* ($T_J$) measures the time between the first video frame being recorded until it reaches the server. This delay is very much dependent on the time a protocol needs for establishing a streaming session. The join time is a lower bound for the latency and it is measured in milliseconds. The rationale behind introducing the join time is that MBS users want to quickly share their videos to an audience. The join time is the lower bound for the video streaming delay until it can be watched by a client.

Besides the initial join time, the different protocols may cope differently with changing upload bandwidths. This is addressed by the current *latency* ($T_L$) which measures the time between capturing a video chunk on the recording device and the chunk is completely received on the server: $T_L = t_R - t_S$.

Continuity Index (CI) represents the quota of a video stream that is stall-free. An optimal streaming session has a CI of one. The CI is represented by: $CI = 1 - \frac{N_{DL}}{N_{UVM}}$, where $N_{DL}$ is the number of the delayed video chunks. Here, $N_{UVM}$ represents the total number of unique video chunks available. Duplicate messages are not considered.

#### 5.2.2.2 *Simulation Setup*

A simulative analysis is chosen to assess the strengths and weaknesses for hundreds of streaming users. The simulative analysis is performed using the Simonstrator platform [**Richerzhagen2015**]. The metrics introduced in Section 5.2.2.1 are used for evaluation. The experiments are repeated 10 times using varying simulation seeds. Depicted figures that include confidence intervals indicate a confidence of 95%. Table 13 shows the simulation setup for the concurrent upload scenario.

*Adaptive Video Streaming*

"Adaptive" integrates adaptive video streaming capabilities and is thus capable of switching between different versions of a video. Recorders transcode three video representations in parallel at bit rates of 500, 750 and 1000 $\frac{KBit}{s}$. An adaptation between video segments is possible every second. The adaptation interval has been proposed in the upload protocol DASH-P [**Seo2012**] as a suitable duration for video segments. All unadaptive upload protocols stream at $750\frac{KBit}{s}$.

The concurrent video upload scenario is based on real traces derived from the MBS YouNow for derived broadcaster sessions between 06/27/2015 and the 07/05/2015. From the traces, session start and end times are derived. Each simulation run represents 24 hours

of operating time of the MBS, and is capped to 1000 concurrent streaming sessions. Different users are assigned to different regions, where each region has a single LTE cell tower as a connection to the streaming destination. This is the bottleneck, as it is limited to an upload bandwidth of up to 50 $\frac{\mathrm{MBit}}{\mathrm{s}}$ that is shared across all recording devices in a region. The delay is modeled to be between 100 and 500 milliseconds to show the robustness of the protocols under high latency conditions. The recording devices are randomly assigned to up to 10 regions, where a region consists of 200 devices maximum.

The adaptive scheme adapts initially to the quick-joining upload protocol. As soon as an upload streaming session is established, the "adaptive" scheme aims for increasing the goodput. A decreasing throughput will lead to an increased rate of stalling. As a result, the adaptive scheme switches to the protocol inducing the minimal overhead to reduce the load on the network.

### 5.2.3 *Results for the Remote Streaming Scenario*

Different upload protocols are evaluated one by one and set into comparison to the adaptive scheme. The aim of each comparison is to improve the evaluation metrics, thus minimizing the *join time*, *latency*, *overhead traffic* or maximizing the *goodput* for each recorder. The average stalling time and effective bit rate for each protocol are evaluated. Figure 32 summarizes the performance metrics of the different MBSs.

#### 5.2.3.1 *Non-adaptive Protocols*

*Join Time and Latency*

In respect to the initial join time, the RTMP, RTMFP and DASH-U protocols are the slowest in establishing a streaming session due to their multi-step join procedure. RTMP uses a three-way handshake that requires multiple messages to be transmitted until the first video segment is sent. DASH-U requires the creation and delivery of a manifest file to the server and the selection of an appropriate bit rate until streaming begins. Both rely on TCP, which itself requires a session establishment procedure. Thus, two uncoordinated join procedures on the transport and the application layer are performed. This initial procedure is slower than the regular request-response behavior of DASH-U. The quickest joining procedure is achieved by DASH-P, as it immediately starts uploading video using HTTP POST requests; it does not negotiate the streaming session, but the underlying TCP session has to be established first. Here, UDP promises to be quicker for initial contact with a node. In general, push-based delivery schemes, without initial handshake procedures (DASH-P), outperform more complex schemes. RTMFP fulfills both UDP support and push-based delivery, but suffers from a complex joining procedure.

The latency in the remaining streaming session is insignificantly different for the protocols, where RTMFP shows a slightly lower session delay. Even though it relies on UDP, RTMFP awaits acknowledgments of each message sent. This artificial delay of RTMFP slows down and mitigates the advantages of UDP. As a result, more efficient packet loss mechanisms shall be investigated that do not generate the higher overhead and more quickly send messages. Latency differences are insignificant, but RTMP shows a slight advantage, due to its efficient scheduling and compact message structures. Pull-based MBSs suffer from an increased latency, as the server has to first request the delivery of the video chunks.

*Overhead*

In Figure 32, overhead costs are normalized [0,1] between no cost and the maximum overhead observed in the evaluation. The chosen normalization allows for retrieving small differences more easily. Especially in relation to the video bit rate, the overhead measured is negligible. Both HTTP-based approaches produce by far the highest overhead. The overhead of pull-based protocols, e.g., DASH-U is significantly different to all others. For the push-based protocols, DASH-P suffers from leveraging HTTP. The headers contain plain text information that is neither streaming-specific nor compressed as, e.g., for RTMP. From the remaining push-based protocols the overhead difference between RTMP and RTMFP is minimal - where RTMFP suffers from coping with increased coordination overhead and packet loss compensation due to UDP.

*Goodput*

Under challenged network conditions, the average goodput of the protocols is essential. DASH-based approaches are less efficient due to their HTTP overhead, including its verbose headers. Under situations with increasing and highly varying error rates, RTMFP outperforms RTMP, due to its flexibility and as it is based on UDP, which allows handling erroneous situations on the application layer in a way unaffected by the transport layer.

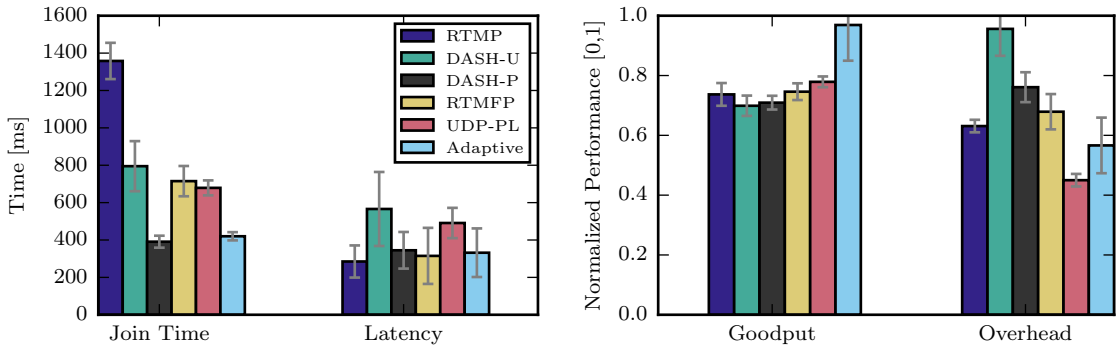



Figure 32: Utility and costs of the different MBSs and the adaptation between the protocols. (a) The achieved join time and delay is depicted, whereas in (b) the normalized goodput and the normalized overhead of the protocols are shown.

#### 5.2.3.2 *"Adaptive"*

The results for using adaptations between protocols are also depicted in Figure 32. Major findings that can be drawn when using an adaptation are a slightly increased overhead and the advantage of always selecting the best protocol available.

*Adaptive Video Streaming*

One of the major advantages of "Adaptive" is the support for adaptive video streaming. It affects both the average bit rate and the stalling rate positively. The average bit rate of the complete video streaming session increases on average by 24.4%, because "Adaptive" can switch to a higher bit rate representation of a video when the throughput conditions are good. At the same time, a low bit rate version of the video may reduce the video quality,

but allow a decrease in the average stall time by around 6.2% to the next best protocol. As a result, the CI is close to one (0.98) for "Adaptive".

*Advantages of the Protocol Adaptation*

As Figure 32 shows the adaptive scheme is always close to the best protocol. Due to its design of switching between existing protocols, the adaptive scheme never significantly outperforms existing schemes. In comparison to RTMP, adapting protocols generate 3.1% more overhead, as duplicate messages and coordination overhead are generated. The adaptations between protocols achieve a similarly low join time as DASH-P, but it saves approximately 9.47% of the overhead. The additional overhead of the "Adaptive" is rather low in relation to the average video bit rate, i.e., 0.89% of the average traffic represent protocol overhead. Only 0.67% of this overhead represents redundant messages or additional coordination operations.

*Switching between Protocols*

In the described scenario, adaptations are invoked every 58 seconds. The average adaptation time is related to the new protocol after the adaptation, and it mainly consists of the join time. Thus, whereas a switch to DASH-P is possible with a negligible delay, the remaining upload protocols need up to 1.4 seconds for establishing a streaming session. By design of the application, an adaptation between two upload protocols is done by running them in parallel until a switch can be achieved.

#### 5.2.3.3 *In Situ Streaming*

The focus of this study lies in the remote streaming case for analyzing the potential of different MBSs and the adaptation between them. All the existing models assume a constantly available device acting as the receiving server, where persistent, mostly TCP-based connections are established. This is contrary to the proposed "in situ streaming" in which spontaneous connections to any device in the vicinity can be established. Due to device mobility and the lack of infrastructure-based networks, frequent bandwidth changes and connection losses can occur. Here, UDP-based protocols such as RTMFP are advantageous. The discussed protocols cannot cope with the lack of central coordination, communication range loss, and an increased device mobility.

### 5.2.4 *Findings of the Study*

The central findings of the conducted study will be summarized, as they are essential to understand the design of LiViU discussed in the next section. It is the aim that LiViU supports varying scenarios, i.e., remote, in-situ, and hybrid streaming, as well as changing application requirements.

UDP offers superior performance in error-free conditions, but even the standardized and widely used protocols such as RTMFP neglect the advantages and show similar issues as the TCP-based protocols, e.g., a slow join time. A novel MBS should leverage the advantages offered by UDP.

The push-based UDP transmission has more superior performance in the concurrent streaming scenario for goodput. RTMFP requires an in-order sending of messages, which are all acknowledged. This adds overhead and leads to additional delay if a message is lost. As soon as the sender does not receive an acknowledgment, the message is sent again.

In general, the scheduling mode (either push- or pull-based) has a higher influence on the performance metrics than the transport layer protocol (either UDP or TCP). Pull-based protocols are suitable when a centrally executed application requires specific chunks of video streams. Switching between these scheduling mechanisms shows promising potential. To conclude the findings for scheduling, different schemes promise to support varying application requirements, as, e.g., pull-based delivery allows a central coordination by a server, whereas push-based delivery supports low-delay streaming.

Two other findings are made that address the joining procedure and content adaptation usage. The joining procedure has a major influence on the liveliness of a video stream. It varies significantly between the protocols and should be designed in a manner to quickly distribute the video stream and avoid long-term joining procedures. Also, none of the protocols leverages adaptation of the video on the mobile device; thus, they cannot cope with changing upload conditions as efficiently as "Adaptive". The simulative evaluation has shown how beneficial adaptive video streaming is.

For the in situ streaming, the number of connections per device is higher, as a single device has to stream to all receivers instantly. Even though it has not been validated in the simulations, the challenges are obvious, and it is clear that no discussed protocol offers solutions.

## 5.3 DESIGN OF A NOVEL MBS

The proposed adaptation between different video upload protocols results in significant costs, as, e.g., the receiver side needs to maintain all protocol stacks in an active state. Derived from the previous study, a novel, adaptive MBS called LiViU is proposed.

### 5.3.1 *Features of LiViU*

LiViU relies on a set of features which are derived from the discussed application scenarios and the conducted simulative study:

DP1. Support for multiple streaming scenarios.

DP2. Support for content and mechanism adaptation.

DP3. Leveraging UDP as a transport layer protocol.

DP4. Support for auxiliary data transport.

DP5. Encapsulation of transmission functionality.

LiViU supports remote, in situ and hybrid streaming scenarios and aims for reliable and efficient video collection. Mechanisms have to be proposed that aim at (1) a high bit rate but rather high delay - e.g., several seconds - for remote streaming as well as (2) low-delay - e.g., milliseconds - and rather low bit rates for in situ streaming. LiViU must be capable to support receivers of both remote and in situ streaming at the same time.

LiViU allows *adaptations* of both the *content* and the *scheduling mechanisms*. *Content adaptation* should be realized using adaptive video streaming. Each device produces multiple representations in parallel. Adaptation of the networks addresses the usage of scheduling mechanisms, i.e., either push- or pull-based delivery. The different scheduling schemes have shown varying benefits depending on multimedia application requirements and environmental conditions. Whereas push-based delivery shows a minimal delay in delivery

and generates reduced overhead, multimedia applications such as video composition benefit from a centrally controlled, pull-based media upload. In relation to the scenario, remote, in situ, or hybrid streaming, the management functions of LiViU have to adapt as well.

As a result of the superior performance for live streaming scenarios, LiViU uses the unreliable *UDP* in the transport layer and compensates its weaknesses by additional application layer mechanisms. LiViU is message-oriented. It copes with the degraded reliability by ensuring the processing of messages in the correct order and integrates error compensation mechanisms.

The protocol shall be able to transport *media streams*, as well as *auxiliary data*, which can be monitoring data such as performance metrics as well as auxiliary sensor data such as locations, accelerometer or other device sensors. Auxiliary data can be leveraged by the multimedia application, which consumes the media streams and is essential if the PaSC (see Chapter 4) is used. The protocols shall be used to transport both media, i.e., audio and video, and auxiliary sensor data.

An overview of the architecture of LiViU is given in Figure 33. It depicts the concepts of *media management*, which addresses all video and auxiliary-data-relevant functionality, as well as the *transmission* functionality, which addresses concepts introduced in LiViU to send and receive data. The yellow boxes depict modules required for the remote (as well as any other) scenario, the gray boxes indicate functionality needed for in situ streaming, which is discussed in Section 5.4.

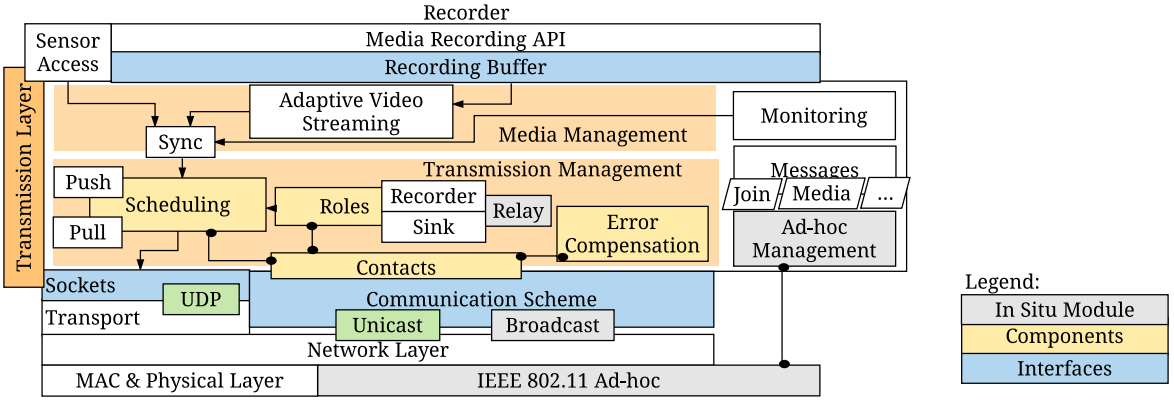



Figure 33: Architecture of LiViU for supporting remote and in situ streaming scenarios.

### 5.3.2 *Video Management*

On the recorder side, the media management is responsible for creating one or more valid media streams.

#### 5.3.2.1 *Audio-Visual Streaming*

LiViU understands media streams as audio-visual data, which are recorded from a recording device's internal sensors, i.e., cameras and microphones. The media streams are delivered in either a single media container or in two media containers independent of each other. The media container encapsulates the media tracks and ensures that the receiver can initiate the video stream playback and an in-order consumption of the video stream. As the visual part of the media stream constitutes a significantly higher portion of the generated data traffic, as well as the computational complexity, the focus for the remainder of this chapter lies solely on video - and thus, video streams.

To enable today's mobile recording devices to stream video in real-time, hardware implementations for compression are used[3]. The support of hardware encoding capabilities allows for considering the H.264/AVC encoding in this work. The concepts can be mapped to the recently proposed H.265/HEVC. Both encoding standards rely on the encoding of a Group of Pictures (GoP), which defines an independently decodable video chunk. Furthermore, as an abstraction from the network, the so-called Network Abstraction Layer (NAL) units are used. They encapsulate video segments, which can be streamed over a network, and independently interpreted by a decoder. Receiving of a single NAL unit does not mean that one or multiple video frames can be decoded, whereas a complete GoP allows the same. Thus, a GoP usually consists of many NAL units, when a video is being transmitted over a network.

A video receiving device needs meta information to setup a video decoder. This metadata produced by the widely supported H.264/AVC codec includes the Sequence Parameter Set (NAL unit type 7) and Picture Parameter Set (NAL unit type 8). The Sequence Parameter Set contains information to understand a sequence of encoded video frames, whereas the Picture Parameter Set defines parameters to understand how an individual frame can be decoded. Both NAL units parameterize the decoder on the receiver side of a stream to be able to decode a GoP without any further information [**ITU-TH2642016**, **ITU-TH2652015**]. This enables a receiver of a video stream to instantly play back or process the video stream - even if parts are lost.

#### 5.3.2.2 *Adaptive Video Streaming*

Adaptive video streaming allows one to record from a single camera and encode the stream in different bit rate representations. During a streaming session, the transmission component decides, what representation should be streamed depending on current network conditions. Until now, this concept is solely available for the delivery of video streams and not on the recording device. Industry solutions as well as research proposals [**Se02012**] do not assume that it is feasible to create multiple representations to instantly switch between them.

This thesis discusses the extension of the media recording API on Android phones to set up different encoding threads on the mobile device. A realization is achieved, as the hardware encoding of current smartphone generations can be leveraged for the transcoding of video. Transcoding means the translation of video attributes (i.e., codec, frame rate, bit rate, resolution) from an incoming video stream to one or many output representations. This is a computationally intensive process. The video codecs are set up with similar parameters but may differ in the target bit rate, frame rate, and video resolution. Each received video frame is handed over to the video encoding thread. As the sequential encoding of different resolutions and frame rates would be too time-consuming on the CPUs of a smart mobile device, the graphics rendering API of Android is used to run the transcoding on the GPU. Each raw video frame retrieved by the camera is converted into a two-dimensional texture, which is represented as a three-dimensional texture for multiple frames. Using a texture, the GPU on the mobile device allows quick manipulation of the resolution and frame rate. The Open Graphics Library for Embedded Systems (OpenGLES) library on the mobile devices is used[4]. Each encoding thread operates on a copy of the texture and manipulates it

3 Supported hardware implementations for mobile devices are defined by the OS, as, e.g., for Android: https://developer.android.com/guide/appendix/media-formats.html; Visited on: 08/30/2016, or iOS: https://developer.apple.com/library/ios/technotes/tn2224; Visited on: 08/30/2016.

4 https://source.android.com/devices/graphics/arch-egl-opengl.html; Visited on: 09/23/2016

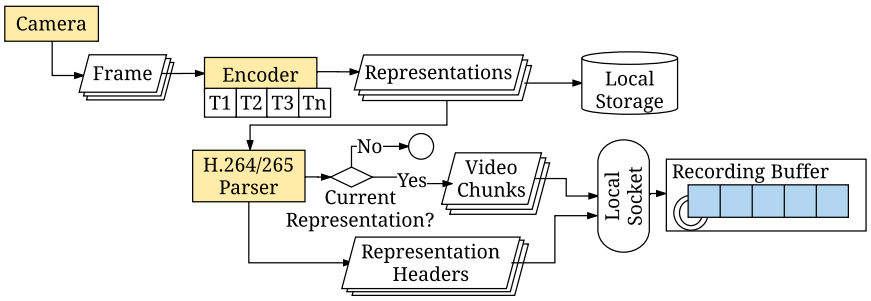



Figure 34: Generation of adaptive video streams on a smart mobile device.

according to the desired frame rate and resolution properties. The final step hands the texture buffer to the respective video encoding object which leverages the built-in hardware to encode the representation at the desired bit rate. The resulting H.264/AVC raw video representations are consecutively written to the recording buffer in the main memory of a smart mobile device.

A real-time capable video parsing service analyzes the consecutively written video files and offers them to the transmission functionality of LiViU. The complexity of understanding when a switch can be conducted without any artifacts on the receiver side is hidden in the parsing service. This is possible, as the NAL unit types 7 and 8 allow for determining the position of independently decodable video chunks. Video chunk boundaries are determined by the GoP. These GoPs can also be quickly identified while parsing the video stream using the NAL unit of type 5 as they represent the start of a GoP [**ITU-TH2642016**, **ITU-TH2652015**]. The proposed approach transcodes a limited number of $N_{Rep}$ representations on the mobile devices during the streaming session in real-time ($N_{Rep} = 4$ on an LG Nexus 5 with 1080p).

Whereas video representations of different bit rates can be easily stitched, the frame rates and resolutions need a mapping by interpolating the dimensions to each other. All resolutions that a mobile recorder generates need to be an integer multiple of the width and height of the lowest resolution. Similarly, the frame rate needs to be an integer multiple of the lowest frame rate.

#### 5.3.2.3 *Auxiliary Data*

Auxiliary data is required by many multimedia applications, especially for the video composition application discussed in Chapter 6. This data consists of monitoring data such as performance metrics (i.e., overhead, goodput, join time and latency) as well as auxiliary sensors. Both describe environmental conditions when recording a video stream.

Auxiliary sensor readings are required for the quality assessment discussed in Chapter 4. Sensors commonly in use are location providers such as the GPS, accelerometer, gyroscope or the light sensor.

The data is stored in individual monitoring and sensor messages and transmitted independent of the video streams. The scheduling of the respective messages can be defined by the application, but it ensures that readings are aggregated and sent at an adjustable frequency. This frequency is chosen to address the requirements of the application, i.e., for just-in-time data processing, and should simultaneously reduce the overhead by sending as few messages as required. All auxiliary data is annotated by timing information, which is required to link the reading to the respective video time.

#### 5.3.2.4 *Synchronization of Streams*

To allow synchronization of the auxiliary data with the audio and video streams, NTP is used. It is assumed that all devices synchronize their clocks using central timing servers [**rfc5905**]. A constant Internet connection is assumed, which can then guarantee accuracy at an error of 10 milliseconds. Synchronized clocks are used to annotate each message and each video chunk with timestamps to allow a resynchronization of video and auxiliary data streams.

### 5.3.3 *Transmission*

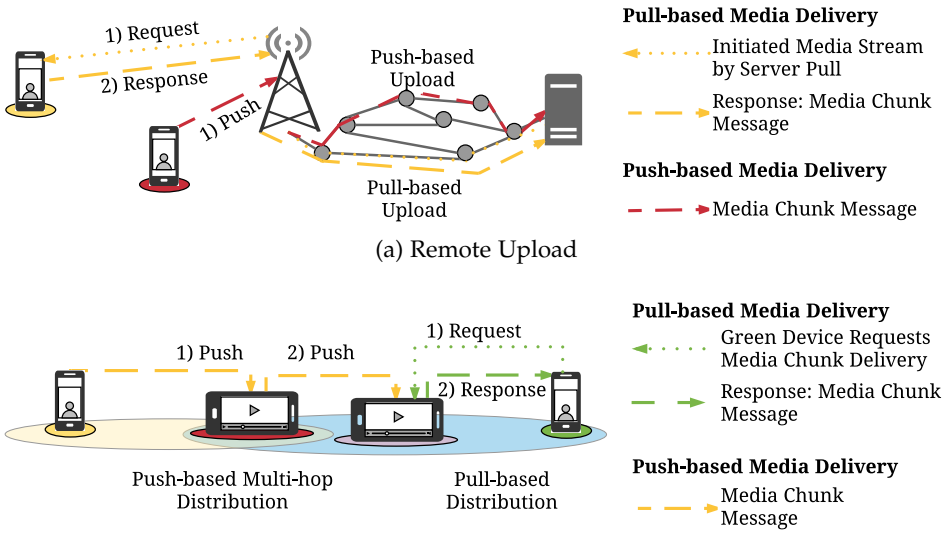
Devices using LiViU record video streams and upload them in a content-adaptive manner by using a reliable transmission layer. This layer offers adaptive scheduling of video streams, coordination of the devices which generates a minimum of overhead, and capabilities to cope with the unreliability of UDP. Derived from the system model - proposed in Section 5.2.1.6 - the contact management module is integrated into LiViU. It runs on each device and allows remote devices to be represented as contacts: a combination of an IP address and a port. The contact types represent the roles of each device in a remote streaming scenario. A device can either be a sender or a receiver of a video stream.

#### 5.3.3.1 *Message Scheduling*

The message scheduling can be classified into the quick stream joining procedure, scheduling type - either push or pull-based - and the timing of message sending.

*Push or Pull*

For different scenarios, a pull-based delivery is beneficial to cope with rapidly changing application needs. LiViU can adapt between different scheduling schemes: push-based




(a) Remote Upload

(b) In Situ Upload

Figure 35: Push and pull scheduling for uploading video streams in remote and in situ streaming sessions.

delivery of media messages and pull-based retrieval of the same (see Figure 35). By default, LiViU uses a push-based delivery to allow a low-delay streaming at a minimal overhead. The switch between two modes can be initiated by a "Pause Request Message" sent by

the receiver of a media stream, which indicates that the default mode - push-based - is stopped. This message includes a hint to reply to the request. As soon as the request is acknowledged by the media recorder, the media receiver requests subsequent media chunks. The pull-based delivery of chunks is controlled by the application on the receiver side, which can determine when to request the appropriate media chunk.

*Quick Stream Joining Procedure*

By default, a push-based delivery is chosen, which allows a minimal join time. Existing protocols rely on long-lasting join procedures and thus increase the join time. LiViU applies quick streaming by assuming that a session can be successfully established without an initial handshake procedure. The join procedure is not executed in a first step; instead, the recording device assumes, that the remote end can be reached, and instantly starts recording and sending video chunks. By using UDP, a connection-establishing procedure as known from TCP is avoided. As the network conditions are initially unknown, the minimal representation is chosen to be streamed to achieve the highest likelihood of timely video stream processing on the receiver side.

The joining procedure is still required to establish a reliable connection state, and to coordinate desired streaming properties such as the resolution frame rate and desired bit rate, as well as encryption or DRM mechanisms. This is achieved as the join procedure is initiated after the first video chunk is transmitted. As no reliable throughput measurements are available before the streaming begins and to avoid congestion, the join procedure is not executed in parallel to the video transmission, but slightly delayed.

*Timing*

The push-based delivery enables LiViU to instantly hand over messages provided by the media recording API to lower communication layers.

Two exceptions exist for timing video and ”leave” messages. Once a mobile recorder indicates that it is time to stop streaming, the ”leave” message transmission is postponed until the remaining video chunks are transmitted. The leave message indicates that the video sinks on the receiver’s side can be closed, and no more video chunks will be sent.

Timing of video messages can be adjusted by a request on the receiver side to either continue or pause the transmission. This indicates to LiViU to switch to a pull-based delivery scheme. Here, the multimedia application on the receiver side determines when to request a video chunk, e.g., requesting only every $n$-th video chunk. Video chunks contain segments that can be independently decoded; this allows for a video quality assessment of parts of a stream (as proposed in Chapter 4) and needed by a video composition system (as proposed in Chapter 6).

#### 5.3.3.2 *Messages*

LiViU is a message-oriented media streaming protocol. Each message consists of a header and a body element. The body contains the payload, which is specific to the message type but can be empty. The header is used by the LiViU protocol to steer the protocol and enable routing on the lower layers (see Figure 36). Each header is designed to generate the least possible overhead, which is also the reason why the headers differ in remote streaming and ”in situ streaming”.

The header of each message consists of at least the sender and receiver contact information. Each LiViU header can also have a notification if a reply is required, and a timestamp

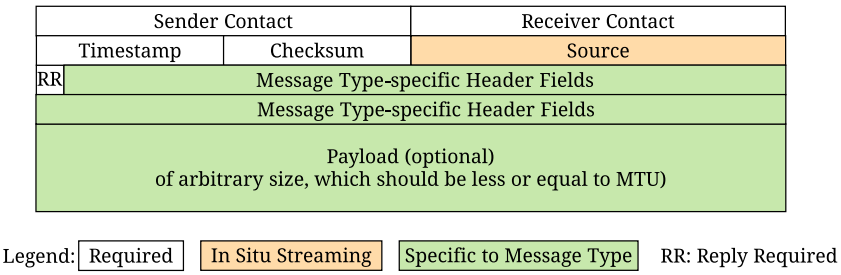



Figure 36: Format of a LiViU message including header fields.

that represents the message creation time. Whereas the sender and receiver contact information are required for communication, the timestamp and reply requests are specific to the LiViU protocol. As the protocol overhead shall be kept low, not all messages are acknowledged automatically. Not discussed further is a Cyclic Redundancy Check (CRC) checksum which indicates the validity of each message. Additional header fields are available depending on the message type. The different message types for a remote streaming session are described below.

*Join Message*

A join message indicates to a video stream receiver that a new streaming session begins. Usually, it is received after the server already received some video chunks due to the integrated quick stream protocol. A join message includes the video information that is transmitted including the representation descriptions, i.e., number of representations, their frame rates, resolutions, and bit rates[5]. Also, a list of available auxiliary sensors is added. Once processed by the receiver, a reply is sent to the sender of the join message to acknowledge that the streaming session is completely established.

*Leave Message*

The leave message indicates that all media files related to a recorder can be closed and state information established on the receiver side associated with the recorder can be deleted. After the leave message is received and processed, no further messages from this sender can be processed.

*Media Message*

The main purpose of LiViU is to transmit video and audio streams to remote receivers. Media messages contain chunks of the media as a payload, with variable sizes. As metadata, these messages contain header information on the streamed representation, the current chunk identifier, and the media type. The representation information is a single integer indicating the representation index. The chunk identifier is unique per device and media type. It is an increasing integer, which indicates the sequence of a media stream. The type of message indicates whether the current stream consists of an audio or video container. Media messages additionally contain information on the latest chunk available on the recorder side to allow a receiver to estimate the delay.

5 When a server receives the first video chunk, it does not have to know the video encoding used as it can parse the initial bytes of a video stream.

*Auxiliary Data Message*

Auxiliary data messages consist of sensor messages and monitoring messages. Only the auxiliary sensor messages are explained here. As auxiliary sensor data is comparably small, if possible multiple samples from a sensor are packed into a single message for transmission. The timestamps indicate when the sensor generated those samples. A type flag indicates what sensor produced the sample. The typical sensors used by multimedia applications are location providers, e.g., the GPS, gyroscope, accelerometer and light sensor.

*Pause and Record Messages*

Devices that are recording a video can decide to stop streaming video to a remote receiver, e.g. if the network conditions are too poor to stream the lowest representation. Pause and record messages allow the client to coordinate to actively use or stop LiViU. Pausing the MBS indicates that no additional messages are sent until a record message is received again. This temporarily disables the scheduling, but keeps track of active contacts. A pause message is necessary if the push-based delivery of a video shall be replaced by a pull-based receiver side retrieval of video. These messages contain no additional header fields.

*Request Message*

As LiViU can operate in both pull- and push-based scheduling, it allows the receiver to request all video chunks individually. Also, the coordination messages - except for join and leave - can be requested. Thus, in pull-based scheduling, the receiver of a media stream would regularly send request messages containing the chunk identifier and a reply request to the recording device. The receiving device determines the rate in which video chunks are requested. Request messages contain the same header fields as the respective message type being requested.

Some special forms of requests are available for controlling the LiViU scheduling. Rate control requests are introduced for video streams to determine at what intervals chunks will be pushed, and which representation index is used.

#### 5.3.3.3 *Coping with the Unreliability of UDP*

UDP is unreliable regarding congestion avoidance, in-order message delivery and packet losses or payload errors. Whereas the avoidance of congestion is out of the scope of this work, our focus lies on the in-order processing and compensation of packet losses. It is assumed that payload errors can easily be detected by a checksum (CRC) transmitted with the message.

The loss of a message can be compensated by encoding the content in a redundant manner in the remaining messages, as, e.g., proposed by Forward Error Correction (FEC), or by solving the problem using a re-request of the messages in a manner as proposed by ARQ. In ARQ, packet losses or errors in messages, e.g., detected via CRC, are compensated using a retransmission of the messages. Also, it ensures in-order processing, as a sliding window is assumed on both the sender and receiver side of messages, which annotates individual messages by a sequence number. Thus, ARQ assumes a duplex channel for communication.

The realization of a selective repeat mechanism for LiViU uses a sliding window on the sender and receiver side. If a message is found that is not received in the correct order - or that is erroneous - subsequent messages are buffered. A negative acknowledgment

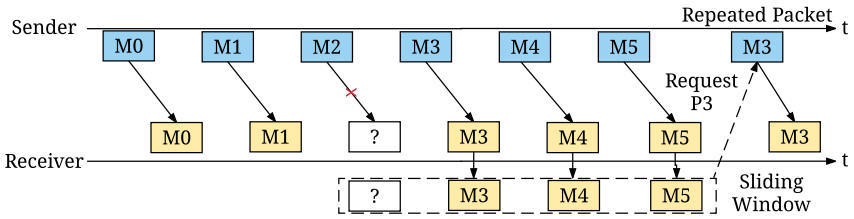



Figure 37: The proposed method for compensating transmission errors using unreliable UDP connection: a request-repeat ARQ scheme.

(request message) is sent to the sender to request the missing video chunk. As soon the message is received correctly, the subsequent, buffered messages are also processed.

Figure 37 depicts the sliding window which can store M packets on the receiver and sender side. The order of the messages is not guaranteed by UDP, so the sliding window reestablishes the order and detects if packets in a sequence are missing or if a message is corrupt. In contrast to other selective repeat implementations, no messages are acknowledged. Only missing or corrupted messages are requested again from the sender. The correctness of a message is validated using the CRC. Missing packets are identified by the sequence numbers created while recording. If a media chunk message with an index $> i$ is received but $i$ is missing, a singleton deadline timer of $D_t + 2 \times \sigma D_t$ is set, after which the chunk is requested, assuming a normal distribution of the delay. $D_t$ represents the last sampled end-to-end delay measurement. This approach ensures a reliable delivery of messages and an in-time and in-order processing of messages. A detected error is compensated similarly to a packet loss by requesting the message again.

#### 5.3.3.4 *Goodput-related Media Adaptation*

The link between transmission functionality and the media management is the video representation adaptation based on the current throughput measurements. From the metrics defined in Section 5.2.2.1, the current application layer throughput is derived. It is used for deciding which video representation to choose, and is thus calculated each time a video adaptation can be executed. Bit rates of the representations are compared with the current throughput, and the representation index ($id_R$) is chosen whose bit rate is equal or slightly below the current throughput:

$$argmax(B(id_R)) \qquad where \quad B(id_R) \leqslant TP_t \tag{28}$$

where $B()$ represents the bit rate of representation $id_R$. It is assumed that a higher identifier indicates a higher bit rate. $TP_t$ represents the throughput measured at time $t$. Independent of the throughput, a multimedia application, e.g., for video composition, can request to deliver a specific representation.

#### 5.3.3.5 *Monitoring*

The transmission layer, especially the media adaptation, requires a continuous and reliable measurement of the end-to-end throughput and delay. An independent monitoring service conducting active throughput measurements is chosen, as proposed by Stohr et al. [**Stohr2014**, **Stohr2016**]. The monitoring service has been modeled, designed and evaluated using LiViU but is not in the focus of this work.

## 5.4 LIVIU FOR IN SITU VIDEO TRANSMISSION

An aim of in situ streaming is a low latency streaming to close-by receivers, who instantly decode and play back the video stream. This section gives insights on additional concepts needed for LiViU to support in situ streaming scenarios.

### 5.4.1 *IEEE 802.11 Ad-hoc Communication*

As an underlay for the LiViU protocol, an IEEE 802.11 network is assumed which operates in Independent Basic Service Set (IBSS) mode [**WLANStandard2010**], called ad-hoc communication in the remaining work. The ad-hoc mode requires no central infrastructure [**Royer1999**]. In IBSS mode the device listens for beacons containing a specified Service Set Identifier (SSID). The SSID defines the unique name of a wireless network. Individual devices use the Basic Service Set Identification (BSSID) for identification. If a device received no beacons with the same SSID, it starts sending beacons to advertise the network.

In case a device receives beacons with the same SSID, it initiates a connection establishment procedure. In order to do that, the device sends a probe request. As soon as it is acknowledged, a time synchronization is initiated. This network merge happens when one group of devices meets another group of devices with the same SSID. A device that receives a probe response will also take over the BSSID of any other station. The device which recently established an SSID takes over the BSSIDs of the older network. A device will start sending beacons periodically when it does not hear a beacon from other devices anymore. As soon as a device returns into communication coverage, it automatically establishes a connection to the ad-hoc network.

The ad-hoc network illustrates the challenges for the design of LiViU. LiViU is designed so that all devices coordinate communication parameters between each other. The limited communication range of each wireless device requires that each device can solely communicate with nearby devices. For communication over larger distances, intermediate devices need to route data to the respective receivers [**Royer1999**]. LiViU copes with these challenges on the application layer.

### 5.4.2 *Device Roles*

Derived from the used network technology, devices using LiViU may have different roles. The roles of recorder and receiver of a video stream are extended by a "relay" node, which receives a media stream and sends it to other interested devices. The role management module allows the instant switch between the roles as well as enables and disables the respective functionalities. Its functionality is linked to the contact management, which keeps track of the sender and receiver addresses. Whereas scheduling uses the contact management to identify the video stream's recorder and receivers, the role management actively triggers scheduling to perform an action.

At the same time, a recorder represents a video recording device, which uses LiViU to distribute a video stream. Recorders retrieve from the media recording API chunks at a constant rate. The video chunks are encapsulated in messages and sent to at least one receiver. At the same time, the recorder is listening to local devices interested in receiving the video stream. A single recorder can thus distribute a video up to $n_S$ nearby receivers, where $n_S \leqslant \frac{TP_t}{B(id_R)}$, where $TP_t$ represents the current bandwidth on the application layer, and $B(id_R)$ the bit rate of the video representation streamed. At the same time $n_S$ should

be chosen in a way that guarantees the computational burden does not exceed the capacity of the recording device.

Receivers retrieve and decode a video stream. At a point in time, receivers have exactly one incoming video stream. Because of device mobility, spontaneous disconnections to a recorder can occur. Thus, the new role of a relay is to retrieve a video stream and redirect it to the receivers. Relays are former sinks, which are not only interested in receiving a video stream for decoding and playback, but also offer the stream to other sinks.

As shown in Figure 33, the role management module is part of the software stack available on each device running LiViU. This role management module is required in the case of in situ streaming only, as users of the devices may quickly switch roles. At the start, users decide on choosing the role of a sender or a receiver. During the streaming session, receivers continuously evaluate whether to become a relay for the received video stream. The concept for scheduling the senders and relays is described in the next section.

### 5.4.3 *Contact Management*

From the perspective of a video recording device, a major change is the switch from a 1-to-1 communication pattern to 1-to-n. Similarly, the role of a relay is new in which a single recorder transmits the same message to multiple sinks. In the contact management module, only senders and receivers are distinguished. The recording device and a relay keep track of the receivers of a stream in the form of an ordered list and the respective delivery mode: direct or using a relay. In contrast to the remote streaming case, this can be a list of multiple receivers.

A central aspect of creating an in situ MBS is the initial connection setup. To achieve this, LiViU applies a continuous heartbeat signaling of actively participating devices in the overlay. As soon as a device starts the LiViU application, it uses a network layer broadcast address to continuously inform (every 5 seconds) other nearby devices on the available video streams.

LiViU maintains connections, even though the receiver leaves the communication range of the recorder, by establishing a multi-hop communication. An association with a video stream is only possible if the devices are in direct communication range with the recorder. The join message has annotations on the respective video stream information. The remaining node types include information on which stream they are receiving and the current location in the form of an accurate location provider, e.g., in outdoor scenarios by GPS. The position information is required by the scheduling and routing maintenance.

### 5.4.4 *Routing Media*

To distribute a video stream to close-by receivers LiViU proposes a novel routing scheme.

#### 5.4.4.1 *How LiViU Routes a Media Stream*

A device recording a video stream regularly publishes announcement messages for advertising the video stream. Newly joining devices can subscribe to an announced video stream. Figure 38 shows the concept of the video chunk dissemination concept applied by LiViU.

A join message for in situ streaming contains the sender's contact addresses as well as the geographic location of the device. To balance the load, a recorder does not directly deliver video streams to all interested receivers, instead it attempts to build a geographi-

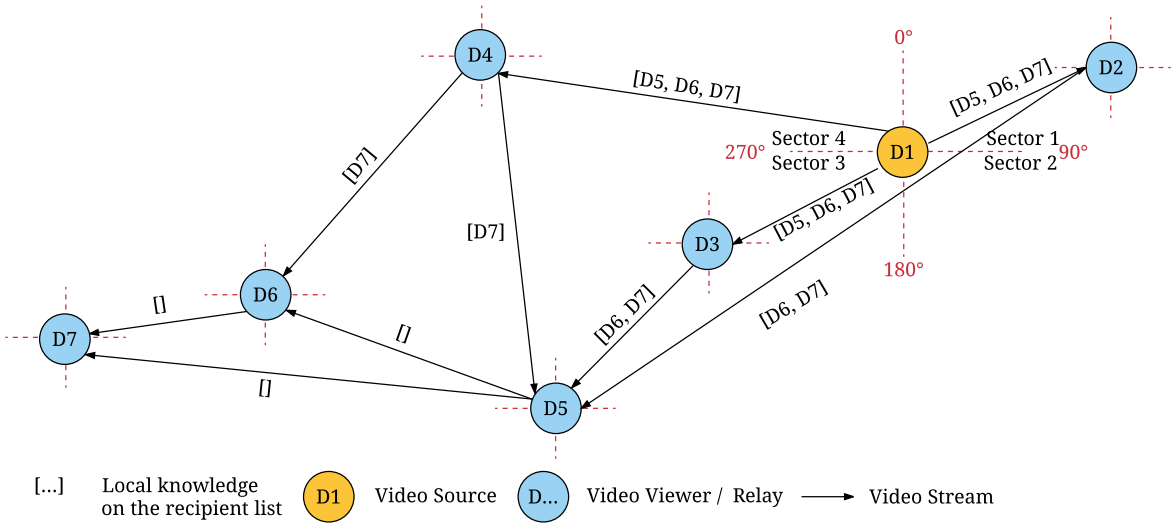



Figure 38: Example for LiViU in situ routing of media chunks in relation to sectors.

cally distributed delivery tree. Figure 38 depicts a typical in situ streaming scenario using LiViU. The recorder determines the relative bearing in degrees (0°, 360°] and the distance to each interested receiver and relay it knows. The bearing indicates the orientation difference to the geographic north; between 0° and 360° is divided into sectors of equal size. Categorizing neighboring devices in an ad-hoc network according to their geo-location is a well-established method, e.g., used for decentralized monitoring systems [**Gross2012**].

In the remaining work, four sectors are assumed with a bearing of (0°, 90°] for sector one, (90°, 180°] for sector two, (180°, 270°] for sector three, and (270°, 360°] for sector four.

A higher number of sectors increases the potential number of first hop deliveries, and thus offers a reduced average streaming delay. The reduction of the number reduces the computational load of the single device. A receiver or relay is selected from each sector based on the closest distance. Each video chunk is distributed to the individual sectors. A message contains a list of the intended receivers - the so-called recipient list. Upon receiving a message, the receiver annotates the message and forwards it in a similar manner as the recorder, i.e., selecting the closest recipient in each sector. If a receiver knows that interested devices are not listed in the recipient list, it adds them.

Position updates, and thus a recalculation of the distribution topology, are initiated with regularly sent advertisement messages. Each device advertises at least its position and the unique video stream identifier it is interested. Also, the advertisement signals a device's online state to all receivers. When an interested node no longer wants to receive the stream, it sends an unlink message to the relay node (see Section 5.4.4.3).

### 5.4.4.2 *Example for Routing Media*

Figure 39 illustrates the video dissemination strategy in a second example scenario. Perfect localization of the devices is assumed, but an imperfection will not significantly harm performance. The devices are represented as $D_0$ representing the recorder and $D_1$, $D_2$, $D_3$, $D_4$ and $D_5$ as devices consuming the video stream. Again, the sector with a bearing of (90, 180] does not contain any interested device and is not considered in the remaining discussion of the example.

A video stream is sent to a first hop from $D_1$ to $D_2$, $D_3$ and $D_4$. Upon receiving and processing, the recipient list solely contains the remaining devices [$D_5$, $D_6$, $D_7$]. In a next step, the device $D_2$ forwards the video streams to $D_5$ in sector (180°, 270°], where the

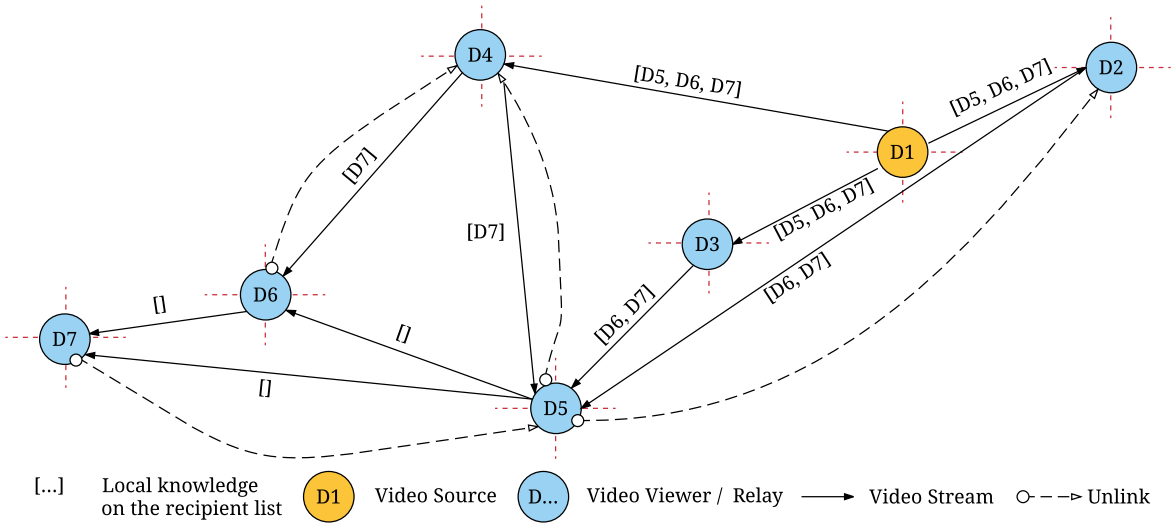



Figure 39: In situ routing scheme of LiViU when using unlink messages to reduce the message overhead.

remaining sectors are skipped. The device $D_2$ forwards the video streams to the recipients in the list: $[D_6, D_7]$. $D_5$ is omitted as it has received the video from $D_3$. The device $D_3$ furthermore processes the recipient map $[D_6, D_7]$. At the same time, $D_4$ forwards a chunk to $D_5$ and $D_6$ with the recipient list solely containing $[D_7]$.

Thus, the device $D_5$ receives three copies of the same chunk at times $t_1$, $t_2$ and $t_3$. It is assumed that $t_1 < t_2 < t_3$. When a video chunk is received at $t_1$ from $D_4$, it is forwarded to $D_7$ with an empty recipient list. At $t_2$, when a chunk is received from $D_3$, it is forwarded to $D_6$ with an empty recipient list. As $D_6$ is closer, the chunk is forwarded only to $D_7$, as the distance is smaller. Finally, at $t_3$, when a chunk is received by $D_2$, it is not forwarded at all.

#### 5.4.4.3 *Linking and Unlinking Devices*

The sector routing prevents unnecessary, redundant routing of video streams to a receiver. Due to imprecise positioning data and the design of the routing, mobility situations can occur when a device receives redundant video chunks from multiple senders.

In such a scenario, the push-based delivery of single devices can be stopped by sending an unlink message. The unlink message, if received, stops any further relaying of incoming video chunks to the sender of the unlink message. On the other hand, the link message allows a device to initiate a new stream from a relay, which receives the stream it wants to decode. The link message acts similarly to a join message, but is instead directed to a relay. Link messages are used if a device runs out of video chunks in the playback buffer. As a result, it is possible that a device temporarily receives chunks from multiple devices.

#### 5.4.4.4 *Reasons for a new Routing Protocol*

Different research groups have proposed routing protocols for ad-hoc streaming. This paragraph describes why LiViU proposes a new approach instead of using protocols such as Ad-hoc On-demand Distance Vector (AODV) [**aodv**] or Optimized Link State Routing (OLSR) [**olsr**]. As major OSs for mobile devices, Android and iOS, have no native support for ad-hoc routing protocols, none of the routing protocols can be run on the lower layers. Any ad-hoc routing protocol has to be implemented and executed on the

application layer. Thus, a potential efficiency increase in comparison to LiViU by running the routing on lower layers is not given.

Also, LiViU aims to support in situ streaming, which implies that the devices may be out of communication range but still in the vicinity of a recording device. In situ streaming implies that the receivers of a stream are on the same event space. Whereas ad-hoc networks have to cope with large multi-hop scenarios, LiViU copes with one or two hops. One implication is that for a small number of hops a timely propagation of routing information to all devices is possible. The efficient propagation of routing information across multiple hops is a central goal of ad-hoc protocols. LiViU's routing protocol is light-weight as each device routes messages without any consultation of other nodes.

As LiViU distributes live video streams, data is sent continuously. It is very likely that a route is used often in a short time, but the proactive establishment of static routes is infeasible due to device mobility. LiViU acts reactively for the video chunks that are transmitted but leverages geo-location information for the routing. In contrast to other reactive protocols such as AODV, no unnecessary coordination overhead by flooding the devices route request messages occurs, and the route finding time is minimized.

One key attribute of LiViU for in situ streaming is risking redundant delivery for the sake of a quick distribution of recorded video chunks. This is contrary to OLSR which informs neighbors on the next hop in order to avoid redundancy. In contrast, for LiViU the receiver of a redundant chunk "unlinks" from its sender to reduce unnecessary data traffic. A video stream's source has not to care about the intermediate hops but only about sending a recorded video chunk to the closest receiver in each sector. Also, more sophisticated routing protocols in ad-hoc networks such as Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.) [**batmanDraft**] do not promise a reduced streaming delay in comparison to LiViU.

For LiViU only devices that are interested in a video stream participate in the network. Thus, no complex coordination is required to motivate devices to participate in the ad-hoc network.

Another feature of LiViU is that each device can set the number of sectors it supports individually. We thereby aim that each device can limit the computational load to its capabilities. The playback and routing of a 1 $\frac{\text{MBit}}{\text{s}}$ video stream can cause a 100% CPU utilization on smart mobile devices [**Halvorsen2008**]. Thus, we avoid to implement one of the protocols and propose with LiViU a protocol for efficient low-delay in situ streaming.

The usage of a broadcast of media chunks is avoided, as strong limitations exist in practice. The smart mobile devices used during the design, implementation and evaluation phase of LiViU limit broadcast transmission to 1 $\frac{\text{MBit}}{\text{s}}$. Even with recent video encoding standards, high-quality videos have a higher bit rate [**Sullivan2012**].

### 5.4.5 *Message Modifications*

In comparison to the previously mentioned scenario of remote stream receivers, the in situ streaming scenario is based on communication with multiple receivers. This results in additional coordination overhead, which is required as the routing of video streams addresses multiple receivers, and the devices can be on the move. The previous sections have illustrated the concepts of allowing LiViU to stream in situ. The remaining section discusses the required message modifications for enabling the in situ communication.

#### 5.4.5.1 *Header Modifications*

As a result, all messages contain a source field that contains the recorder address. The source field in comparison to the sender address depicts the origin of a video stream, whereas the sender is different if a receiver retrieves the video from a relay.

#### 5.4.5.2 *Ad-hoc Messages*

All devices keep local knowledge on their neighboring devices. The neighborhood is defined as all devices in communication range.

*Advertisement Message*

To know which devices belong to the neighborhood, an advertisement message is periodically sent every $T_{BC,A} = 5$ seconds. It contains the role of the device and the identifier of the stream, which is currently recorded or received. The advertisement is sent by each device via broadcast. It allows retrieving which device is active, providing or consuming video.

*Link and Unlink Messages*

Two additional messages are required for the coordination of LiViU devices: the link and unlink messages. These messages are sent to indicate that a device shall start or stop relaying a video stream to the requesting device. The message is acknowledged by the receiver to indicate to a device that no upload capacity is left. Unlink messages are used to reduce redundant delivery of messages.

## 5.5 SUPPORTING DIFFERENT SCENARIOS

The combination of mechanisms for remote and in situ streaming allows LiViU to support multiple scenarios. It is assumed that the smart mobile devices support cellular networks and WLAN in parallel. The remote streaming is using the cellular network, whereas the in situ communication is realized using the WLAN. From a recording device perspective, the contact management module allows to distinguish which contact is remote and which is reachable in ad-hoc communication. As a result, LiViU supports hybrid streaming scenarios in real deployments, if two network interfaces can be used in parallel.

## 5.6 EVALUATION

In this section, the performance of the novel MBS LiViU is evaluated using a prototypical evaluation in two different, realistic streaming situations. The first streaming situation analyzes the remote streaming case over cellular networks, whereas the second performs an analysis of in situ streaming.

### 5.6.1 *Evaluation Setup*

The evaluation consists of different runs repeated under similar conditions. The experiments are realized using 13 different devices from different smartphone generations, split into two evaluations: remote and in situ streaming The mobile devices and network connection settings are shown in Table 14.

The evaluation is conducted to assess LiViU's performance regarding goodput, overhead, duplicate messages, join time, and video stream continuity (CI), as described in Section 5.2.2.1.

Table 14: Device setup for evaluating LiViU.

| | Scenario | Count | Network |
|---|---|---|---|
| LG Nexus 4 (N4) | Remote | 2 | 3G |
| LG Nexus 5 (N5) | Remote | 3 | LTE |
| Samsung Galaxy S6 (S6) | Remote | 1 | LTE |
| Samsung Galaxy S7 (S7) | Remote | 1 | LTE |
| OnePlus One (OPO) | In situ | 6 | 802.11, LTE |

#### 5.6.1.1 *Remote Streaming*

For the remote streaming evaluation, three LG Nexus 5, two LG Nexus 4, one Samsung Galaxy S6, one Galaxy S7 and six OnePlus One are used to assess the performance of LiViU. Video streams are transmitted using the cellular network of Deutsche Telekom in Darmstadt. The remote streaming scenario was evaluated at different PoIs in Darmstadt: a university building, Herrengarten, and Marktplatz. In any scenario, the movement is limited to pedestrian speed. All devices are in the same communication cell and share the bandwidth. The setup consists of a heterogeneous device set connected to different network types, with varying performance. Additional traffic by other devices was not considered. The recording side buffer is set to 50 MB to compensate for situations where the recorder captures video faster than the network can transport it. All devices use synchronized clocks achieved by using NTP with a single clock server.

As a streaming end-point, the "streamlet.de" server is hosted in a data center near Nürnberg, Germany. Each run lasts for approximately twenty minutes, where the initial five minutes leverage the available throughput of the LTE network, and the remaining throughput is shaped on the "streamlet.de" server according to an upload trace of the MBS YouNow from 06/27/2015.

The videos are recorded and then encoded into three representations at a resolution of 720p at 3 $\frac{\text{MBit}}{\text{s}}$, 1.5 $\frac{\text{MBit}}{\text{s}}$ and 750 $\frac{\text{KBit}}{\text{s}}$ at 30 FPS. The GoP length is set to 15 frames. Thus, an adaptation could occur every half second.

All experiments are repeated five times. The focus of the remote streaming evaluation is illustrating the advantages of adaptive video streaming.

#### 5.6.1.2 *In Situ Streaming*

For evaluating the in situ streaming scenario, OnePlus One devices were used, as the other devices do not allow to establish IEEE 802.11 ad-hoc connections. The OnePlus One devices allow in ad-hoc mode using the IEEE 802.11b standard with at most 11 $\frac{\text{MBit}}{\text{s}}$ of physical layer bandwidth. They can reliably stream in ad-hoc mode, without any undesired connection resets. Six devices were used in a stationary device setup and in a mobile setting. At any time, exactly one sender is active. LiViU has been tested to support multiple senders in close vicinity, too. A single limitation can be the capped physical bandwidth of the used IEEE 802.11 standard.

Similar conditions as for the remote streaming scenario are used. If not stated otherwise, the produced video representations are encoded at 480p at 250 $\frac{\text{KBit}}{\text{s}}$, 500 $\frac{\text{KBit}}{\text{s}}$ and 750 $\frac{\text{KBit}}{\text{s}}$ at 30 FPS. The buffer is configured to store up to 250 milliseconds of the video. It is ensured that video chunks of approximately 100 milliseconds can be decoded independently.

All experimental runs are repeated six times with a similar setup, where the in situ streaming is evaluated in a stationary and a mobile setup. The focus in both setups lies on the evaluation of the network characteristics of the system. Note, that for the mobile setup the same environmental conditions as in the stationary one cannot be guaranteed.

*Stationary Setup*

A stationary setup is chosen consisting of six mobile devices with stable positions and within IEEE 802.11 communication range. Each evaluation run is performed for 10 minutes. Figure 40 shows the device setup and intended communication between devices. The experiments are performed in a closed room so that no valid position information can be retrieved. Thus, we manually set the location coordinates in the OS.

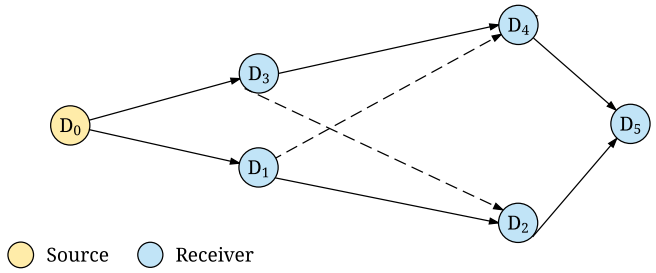



Figure 40: Static topology used for the evaluation of LiViU in the in situ scenario.

The stationary setup is chosen to conduct a parameter study on both the effect of the streaming bit rate on the effective latency and the influence of an increasing number of devices. The evaluation is conducted at different bit rates - from 250 $\frac{\text{KBit}}{\text{s}}$ over 500 $\frac{\text{KBit}}{\text{s}}$ to 750 $\frac{\text{KBit}}{\text{s}}$ - to study the effect of bit rate on performance. Also, the number of devices receiving a stream varies from 1 to 5. In this second experiment, the bit rate is kept constant at 500 $\frac{\text{KBit}}{\text{s}}$.

*Mobile Setup*

In a mobile setup, the in situ streaming functionality is evaluated in six repetitions of similar movements in a limited area. In a user study a perfect reproduction of movement patterns, and the environmental conditions, could not be achieved. The experiments consist of 18 minutes per repetition, which is split into three phases of six minutes each. During the phases, different mobility patterns are evaluated. In the first phase the movement of solely the receivers is evaluated, where the sending device holds a central position. The second phase includes stable positions of the receivers, whereas the sending device is in motion. In the last phase, all devices are in free motion. Due to the mobility of the devices, connections can be lost at any phase. All motion is at pedestrian speeds, thus $\leqslant 7\frac{\text{km}}{\text{h}}$. A fixed bit rate of 500 $\frac{\text{KBit}}{\text{s}}$ is utilized in this scenario. During this evaluation, locations and movement speeds are calculated using GPS. Figure 41 depicts an overview of the evaluation space for the mobile setting and depicts the second phase of the described scenario, in which the sender continuously moves. It indicates that potentially disturbing 802.11 access points may limit communication between the devices. The illustrated access points are not supporting the device-to-device communication, but act as disturbing elements, as their provided networks compete with ad-hoc networks on the physical bandwidth. Thus, in this realistic setup, access points can lead to increased packet drops or connection losses in the in situ, ad-hoc communication.

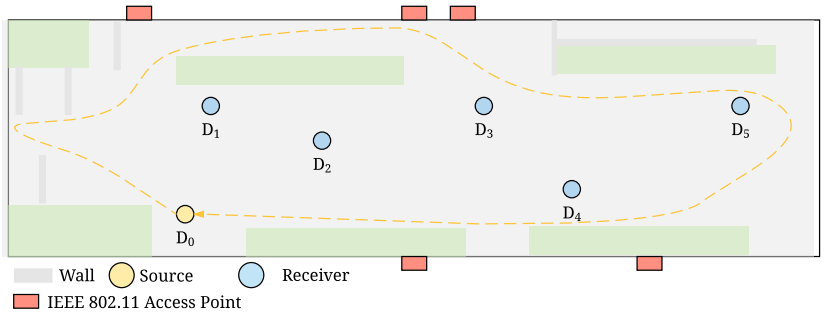



Figure 41: Sketch of the evaluation setup for evaluating LiViU's performance considering mobility. Figure shows phase 2 of the evaluation in which the sending device is in motion.

### 5.6.2 *Performance for Remote Streaming*

The remote streaming scenario is evaluated in cellular networks with the aim to illustrate the advantages of adaptive video streaming concepts in an MBS. To indicate the performance in comparison to other MBSs the initial join time, the continuity of a video stream, the goodput, and the overhead of a streaming session are discussed.

#### 5.6.2.1 *Effect of Content Adaptation on Join Time*

Part of the join time is the preprocessing of the recorded video for transcoding into different representations, the processing of the LiViU application, the transmission and the processing on the server. The delay is measured from capturing a complete video frame until its storage on the remote server. The focus lies on the discussion of the processing times on the mobile device running LiViU as well as the buffer size on the LiViU server. Transmission delays invoked by the cellular network are not discussed in detail, as LiViU does not influence them.

*Transcoding Adaptive Video Streams*

The transcoding process ensures that the lower bit rate representations are available first, whereas encoding at the highest bit rate lasts longest. The time between the first and the last representation is available depends on whether recent H.264/AVC encoding hardware is used. Figure 42 (a) gives an overview of the difference of leveraging software and hardware encoding. Whereas for hardware-support (HW) the time difference between the first and the last representation is below a second, the software-based encoding (SW) requires 8.41 times longer to encode all three representations. Even the hardware encoding needs an average of 0.61 seconds to have a video frame encoded in all three representations, and thus enable the receiving device to perform an adaptation. Thus, as soon as a video chunk is encoded at the lowest representation, it is transmitted to the server. A switch to a higher bit rate representation can be performed as soon as throughput conditions are suitable, higher bit rate chunks are available. Even under good throughput conditions, an initial upload of the lowest bit rate representation is performed to minimize join time.

*Minimal Media Chunk Size*

The size of video chunks being transmitted has an impact on the delay until a stream can be consumed on the receiver side. At the highest bit rate (3 $\frac{\mathrm{MBit}}{\mathrm{s}}$ for remote streaming), a LiViU message could transport around 3 milliseconds of video, and at the lowest bit

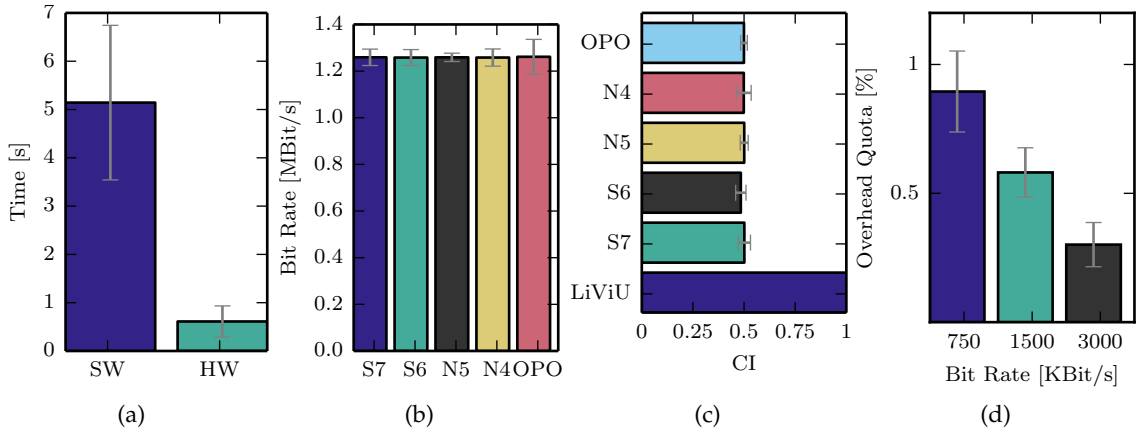



Figure 42: Overview of the performance when using LiViU for a remote streaming scenario. (a) Transcoding speed for adaptive streaming representations on the video encoding hardware (HW) in comparison with software encoding. (b) Average video bit rate using LiViU in the given scenario. (c) Stalling-free upload using LiViU (CI=1) in comparison with non-adaptive streaming approaches on different devices. (d) Percentage of overhead when using LiViU in comparison to the representation bit rate.

rate representation around 12 milliseconds. Independent of a segment length the minimal video chunk to be sent should be chosen wisely on the basis of the packet size that can be sent using UDP. There is a trade-off between immediate transmission, encoding efficiency, and minimum bit rate. When video chunks to be transmitted are smaller than the packet size, the overhead of packet headers will increase detrimental to the goodput. In the remote streaming scenario, a packet size of up to 950 bytes is chosen. At a frame rate of 30 FPS, this requires that the minimal bit rate of a representation $\geqslant 228\frac{\text{KBit}}{\text{s}}$. Any bit rate lower than $228\frac{\text{KBit}}{\text{s}}$ would create an unnecessarily high number of messages, which are only partly filled. The delay is sacrificed to obtain efficient delivery, which keeps the overhead of messages and headers low.

*Remaining Components of the Join Time*

Remaining operations of the LiViU protocol accounted for 119.78 milliseconds on average, where a single outlier required 692.46 milliseconds. It could not be clarified what caused this huge increase. The cellular network caused an additional delay of 79 to 103 milliseconds for LTE. As a result, the delay on the remote streaming side is around 1.7 seconds on average, where a maximum delay of 3.8 seconds has been observed. Delays related to the serialization of the messages, buffer and main memory operations of the protocols, and the storage of the stream on the hard disk are not discussed in detail.

### 5.6.2.2 *Effect of Content Adaptation on Continuity and Goodput*

The protocol overhead is negligible, as the content adaptation and throughput estimation is solely conducted on the client with local resources. No coordination is required for the content adaptation, as throughput measurements are offered by an independent monitoring service [**Stohr2014**, **Stohr2016**].

Under similar conditions, yet with competing devices, the goodput of the adaptive streaming upload achieved 1259.1 $\frac{\text{KBit}}{\text{s}}$ on average with a standard deviation of 7.554 $\frac{\text{KBit}}{\text{s}}$. The differences in repeated runs were mainly caused by uncontrollable environmen-

tal changes, such as, e.g., slight variations of the throughput. They are shown for the five different device types used in the evaluation in Figure 42 (b).

At the same time, the adaptive video streaming concept on the uploading side of a smart mobile device helps to avoid stalling. The stall time would have been enormous if no adaptive video streaming approach had been chosen. The achieved CI of 1 is compared with the different devices in Figure 42 (c). It can be seen, that when streaming constantly at 3 $\frac{\mathrm{MBit}}{\mathrm{s}}$ the CI decreases to between 0.4841 and 0.5014 if no adaptive video streaming is used.

The adaptation could achieve consistent streaming without stalling, requiring 9 to 11 adaptations per 10 minutes of video streaming. All devices streaming in parallel have shown a similar adaptation behavior at similar points in time. This available throughput drop is caused by the used network trace.

#### 5.6.2.3 *Overhead*

Per 10 minutes of streaming, the quota of the number of control messages to the total number of messages is below 0.12% on average, where the size of these messages in comparison to the total traffic generated accounts for only 0.027% (see Figure 42 d). This omits the overhead caused by the message headers. If the lowest bit rate representation (750 $\frac{\mathrm{KBit}}{\mathrm{s}}$) is considered, the total control message traffic as well as the message header, accounts for less than 0.9% of the total traffic.

### 5.6.3 *In Situ Streaming Results*

The in situ streaming under challenging conditions, i.e., with mobility, is discussed regarding the bit rates of the streamed video, the effect of mobility on the continuity of the stream as well as the overhead caused by the decentralized organization of the devices.

#### 5.6.3.1 *Influence of the Representation Bit Rate*

In the stationary setup evaluation, the effect of the video representation's bit rate is evaluated in respect to different performance metrics. The influence of the bit rate is given for CI, the protocol overhead regarding the number of messages, and the delay. As a result, the CI for this stable condition stays nearly constant, variations below 1% are observed. The minimum CI still achieves a continuity of above 99%. A low CI indicates that the protocol is not capable of delivering media chunks in real-time. Rather hard constraints are given for the recording buffer, which stores approximately 250 milliseconds of the received video stream. If packets are lost due to collisions, LiViU will request and has to receive the respective video chunks within this window to avoid stalling. It is obvious from Table 15 that the average and maximum delay do not always allow an in-time delivery of the media stream within the receiver's buffer capacity.

Table 15: In situ streaming: Performance of LiViU for varying bit rates.

| | CI [%] | | | Delay [ms] | | | Overhead [%] | | |
|---|---|---|---|---|---|---|---|---|---|
| Bit rate [$\frac{\mathrm{KBit}}{\mathrm{s}}$] | mean | min | σ | mean | max | σ | mean | max | σ |
| 250 $\frac{\mathrm{KBit}}{\mathrm{s}}$ | 99.87 | 99.74 | 0.041 | 62.9 | 101.54 | 16.64 | 1.33 | 1.361 | 0.028 |
| 500 $\frac{\mathrm{KBit}}{\mathrm{s}}$ | 99.83 | 99.62 | 0.091 | 80.13 | 117.06 | 21.19 | 1.52 | 1.65 | 0.08 |
| 750 $\frac{\mathrm{KBit}}{\mathrm{s}}$ | 99.67 | 99.41 | 0.11 | 175.03 | 315.86 | 98.42 | 1.79 | 1.91 | 0.07 |

The delay increases with the bit rate. This indicates throughput limitations of the IEEE 802.11b network, as the maximum effective bandwidth is achieved at a bit rate of $500\frac{\mathrm{KBit}}{\mathrm{s}}$. The control overhead quota indicates a slight increase to 1.79%. The quota has been chosen to be based on the number of messages and not on the message size, as the control traffic accounts for less than 0.2% of the total traffic.

This is reliant on the given environmental conditions as well as the number of devices participating in the network. Especially in the 2.4 GHz 802.11 network, a significant increase of in situ devices will limit the bit rate. The limitation is given by the physical medium and the processing and scheduling capabilities of the devices. The total join time in the case of in situ streaming is calculated from the availability of a video frame until its playback on the receiver device for a single hop and is 817.83 milliseconds on average ($\sigma = 31.88$ milliseconds). In general, LiViU offers robust streaming under highly changing video bit rates.

#### 5.6.3.2 *Influence of Increasing Interest*

If, during a session, devices join and leave the network, performance metrics may by affected. Therefore, the topology depicted in Figure 40 is set up by joining the devices in a specific order, and leaving the network in the reverse order. From a single node offering the video stream to the topology depicted in Figure 40, the topology is built up as shown in Figure 43. Each topology stage is kept for two minutes to achieve stable results. The main

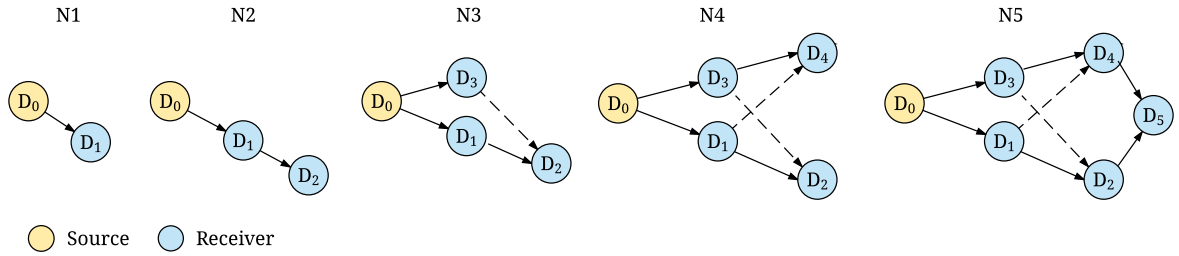



Figure 43: In situ streaming: Performance of LiViU with changing device numbers.

finding of the experiments is that LiViU achieves a stable CI, which is not reduced until step 3, when $D_2$ can receive video chunks from multiple devices. The communication of both $D_1$ and $D_3$ leads to an increase of packet losses or error(s) in the packets, and thus a reduced continuity index. It decreases slightly for N3 at an average 99.91% to N4 at 99.81% and stays stable in N5 at 99.83%, which represents the complete topology.

Similarly, the topology induces a pattern for the control overhead and the quota of redundant chunks received. For the overhead, the maximum is reached at N3 at a control overhead of 1.81%, and in the remaining rounds it is either stable or it declines. As the overhead is calculated in relation to the total number of packets, a peak is reached when most devices are in close range and need to coordinate themselves. Redundant video messages are received when a new device joins the network to quickly retrieve the stream - and if uncoordinated senders in close vicinity to each other do not know of each other as the distribution tree becomes deep, as e.g., for $D_2$, $D_4$ and $D_5$. Due to the proposed routing in LiViU which sends messages based on the distance and bearing between devices, no clear indication is available, which device should deliver video to $D_5$. As long as the receiver does not unlink from one device, it will send video chunks. Whereas the redundant video chunk rate increases for $D_5$ up to at a maximum of 48.74% and in average 23.29%, the pro-

posed unlinking coordination by the receivers limits to a maximum of 8.2% of redundant messages.

#### 5.6.3.3 *Influence of Mobility*

In the second experiment, the mobility of devices (in three phases) is evaluated, consisting of solely receiver mobility, sender mobility only, and a combined evaluation if all devices move.

LiViU's reliability and costs are assessed in environments where devices leave the communication range due to mobility, delays vary, and streaming quality as measured by CI degrades. Also, different IEEE 802.11 access points compete with the ad-hoc network established by LiViU.

Table 16: In situ streaming: Performance in the mobile scenario. Phase 1: Receivers are moving, Phase 2: Sender is moving, and Phase 3: all devices in motion.

| | Phase 1 | | | Phase 2 | | | Phase 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| Run | mean | max | min | mean | max | min | mean | max | min |
| Delay [ms] | 116.57 | 172.71 | 79.8 | 98.86 | 174.26 | 41.35 | 138.56 | 188.29 | 94.77 |
| CI [ms] | 97.66 | 99.82 | 94.15 | 98.81 | 99.9 | 95.65 | 96.74 | 99.32 | 91.25 |
| CO [%] | 4.03 | 5.54 | 1.81 | 2.99 | 7.67 | 0.93 | 6.05 | 11.49 | 2.34 |
| Speed Receivers [$\frac{m}{s^2}$] | 0.811 | 0.927 | 0.599 | 0 | 0 | 0 | 0.917 | 1.038 | 0.735 |
| Speed Sender [$\frac{m}{s^2}$] | 0 | 0 | 0 | 0.974 | 1.03 | 0.924 | 0.959 | 1.04 | 0.895 |

The results, including the speeds of the devices, are shown in Table 16. Mobility has a significant impact on the CI, which in the case of all devices in motion drops to a minimum of 91.25%, where the average in the session is 96.74%. Still, in spite of potential connection losses and re-routing requirements in the scenario, it is ensured that $\geqslant$ 90% of the stream is received and played back in time. The delay from recording until playback is $<$ 650 milliseconds and thus comparably faster than infrastructure-based approaches with delays around 1 second [**Dezfuli2013**]. The increased control overhead (especially in phase 3 where all nodes are moving) depicts that the required coordination leads to more messages being exchanged.

In the mobile scenario, the likelihood of redundant delivery of video messages from different senders to the same receiver is high. The receiving devices can determine, which devices shall stop sending using unlink messages. This mechanism ensures that the duplicate chunk ratio reduced to in average 1.9% at a maximum of 6.03%.

### 5.6.4 *Discussion*

The proposed MBS LiViU achieves efficient streaming both to remote and close-by devices. For remote receivers, the benefits include reliable streaming while coping with rapidly changing throughput conditions, a quick connection setup, and adaptation to different scenarios. The in situ streaming achieves minimal delay between devices while independently organizing the streaming topology at the cost of redundant messages. The evaluation shows that reliable streaming can be achieved despite mobility.

In comparison to other work [**Dezfuli2013**, **Niraula2009**], LiViU achieves a lower delay with considerably less overhead. Furthermore, Niraula et al. show for mobile scenarios that stall-free streaming is possible at a bit rate of 128 $\frac{KBit}{s}$, which is lower than the bit rate that LiViU supports (500 to 750 $\frac{KBit}{s}$). Furthermore, adaptive video streaming is supported by LiViU on the mobile device. In contrast to the work of Seo et al. [**Seo2012**], the proposed system achieves a reduced delay with similar video recording settings, but

in cellular networks. Whereas, the proposed LiViU system achieves a reliable streaming of 720p content in cellular networks without any control of the network, Seo. et al. [**Seo2012**] ran their experiments in private IEEE 802.11 networks without competing traffic. Thus, for high resolutions of up to 720p and non-recent devices, live streaming was not possible. In contrast to their approach, LiViU achieves in-parallel encoding of three 720p videos in real-time, as well as their delivery to remote servers and close-by devices. The availability of different video representations allows adaptation according to the network conditions smoothly. It can be concluded that LiViU achieves reliable video streaming in the different streaming scenarios.

## 5.7 CONCLUSION

This chapter introduced the content- and mechanism-adaptive LiViU, which allows for the efficient delivery of video streams to both remote and close-by receivers. The mechanisms supporting an efficient upload are gathered in an initial simulative study investigating existing MBS protocols. It is shown for remote streaming, that UDP-based protocols in combination with quick application-layer join procedures are most favorable to achieve low join times. The initial join time is critical, as it defines a lower bound for the liveliness for the remaining streaming session. The scheduling of the video stream messages can be either push- or pull-based. Which scheduling should be favored depends on the context, but especially on the application.

From the findings gained in the simulative study, the novel LiViU system has been designed. In the remote streaming case, LiViU achieves high bit rate video streaming with minimal overhead. The proposed protocol includes an on-device transcoding of different video representations, and brings adaptive video streaming to the upload side. Content adaptation in LiViU ensures high continuity of the streaming session without stalling. It is supported by a scheduling mechanism adaptation, which allows LiViU to specify which parts of which video stream shall be transmitted using push- or pull-based delivery. This functionality is offered to the multimedia application. This allows a high flexibility for the proposed multimedia applications in this thesis (PaSC and video composition), but also for future developments.

For in situ streaming scenarios, LiViU achieves a reliable streaming experience even under motion without infrastructure support. In the in situ streaming case, LiViU pursues a geographic distribution of the streams avoiding unnecessary redundancy in the message delivery. As a result, with LiViU a superior MBS is proposed, which is used for the PaSC proposed in the previous chapter as well as the delivery of media streams for video composition, as discussed in the next chapter.

# 6 VIDEO COMPOSITION

LiViU leverages the concept of video and network adaptation to both increase the quality of a video upload session and limit the generated data traffic. Still, LiViU is not capable of mitigating recording quality degradations in a single video stream. Another form of content adaptation, the video composition, can compensate for these quality degradations by always selecting the best parts of different video streams. Video composition assumes that multiple video streams are generated in parallel. A video composition application analyzes different views of a scene and fuses them into one single video stream, depending on the video availability and quality. At any given moment, only one source view is selected to be streamed to the receivers.

A video composition as a form of adaptation between different video views has several advantages. Live UGV is often rather short in duration and cannot cover an entire real-world event in a single source video [**Stohr2015**]. Video composition can ensure that at all available videos are leveraged to retrieve a complete coverage of an event.

Also, the composed video stream can be of a stable quality as the highest quality parts of all source videos are used. Video stream quality is determined by the quality assessment algorithms proposed in Chapter 3 and Chapter 4. If we assume that quality assessment steps are realized on the mobile device during recording, a video composition application can further reduce the generated data traffic as only the high-quality video streams need to be uploaded.

This chapter proposes a combination of a semi-automatic composition that leverages human knowledge to compose videos near real-time and train a novel, automatic video composition system. The basis for our video composition application is a filter stage, which ensures that high quality video views are considered for composition. This thesis proposes two composition applications: a semi-automatic and an automatic composition. The semi-automatic composition relies on crowdsourcing minimal tasks to a group of distributed users and leverages system support to ensure a timely composition. This composition - as human assessment is involved - copes easily with different video genres and content. Derived from manually generated composition models, AutoCompose allows for a fast, automatic composition of video streams. AutoCompose is based on the machine learning mechanism Support Vector Machine - Hidden Markov Model (SVM-HMM), a sequence-tagging machine learning approach that can efficiently compose videos.

The concepts and evaluation results presented in this chapter are a revised and extended version of our peer-reviewed publications [**Wilk2014c**, **Wilk2015c**, **Wilk2015d**].

## 6.1 CONCEPT OF THE PROPOSED VIDEO COMPOSITION

Automatic video composition aims at creating a single video stream (called composed video) from the abundance of a scene recorded in different video recordings. The composed video includes the most desirable video segments from different recordings [**Shrestha2010**]. Figure 44 shows the automatic video composition and discusses the two central questions which are answered in a video composition:

1. Which video view will be selected next for the composed video?

2. When will a switch from one view to another be executed?

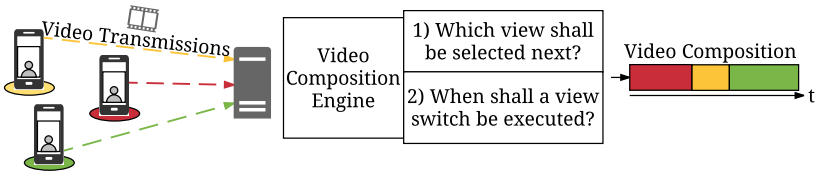



Figure 44: Tasks of automatic video composition algorithms to receive several in-parallel recorded video streams and select one at any given moment for the output.

The first question addresses a point that the composition application has to understand what kind of a scene is recorded, and select the next appropriate video view. This also includes that the continuity of the composed video shall not be broken. The second question discusses that a switch from one view to another is placed in a manner to not degrade the viewing experience or distract viewers. Answers to these questions have to be given for live video streams in real-time and to scale automatically with the increasing amount of UGV.

In Section 2.5.1.2, we discussed how quality is perceived in composed videos. It is commonly agreed that a composed video can achieve a higher perceived quality in comparison to a single video view. Video composition needs to consider the concepts of event coverage, event continuity, the quality-aware selection of video views, the diversity of the selection and cinematographic rules. This chapter discusses these aspects in three different modules: 1) Filter stage, 2) CrowdCompose and 3) AutoCompose.

The *filter stage* preselects videos that should be considered for composition. Not considered are video views at a given point in time that either suffer from a significantly degraded quality or conflict with a cinematographic rule.

*CrowdCompose* is a semi-automatic composition application, which is explained in Section 6.3. It is a semi-automatic approach, as a group of humans make composition decisions, i.e., which view will be selected next and when to switch to it. To enable a cost-efficient and especially quick decision making - complying with real-time constraints - crowdsourcing is chosen as a paradigm. In this context, crowdsourcing implies the design of small and well-defined tasks that can be completed in a short time by a wide range of remote users. Tasks are provisioned to an anonymous group of people using a mediating web platform, and users are compensated for completing the tasks by micropayments. The results of CrowdCompose are a composed live video stream and models that can later be used by AutoCompose to ”learn” human directing styles.

*AutoCompose* learns the composition model proposed by CrowdCompose and thus mimics human composition styles. As video composition relies on an understanding of the decisions in a temporal dimension, i.e., current and previously selected views determine which view is selected next, a sequence-tagging machine learning algorithm relying on the SVM-HMM is used. It conducts video compositions completely free of any human interaction. CrowdCompose and AutoCompose are interchangeable modules, where only one is used at a time.

## 6.2 FILTER STAGE

The filter stage is executed before the composition decisions are made by CrowdCompose and AutoCompose. It determines which views do not comply with the quality constraints as well as cinematographic grammar rules, and removes them from further consideration.

It is set up as a sample-based sequential pipeline. Sample-based implies that the filter stage inspects a video view at two points in time, $t_0$ and $t_1$, where $t_0 < t_1$, the filter stage assumes if the sample at $t_0$ represents the quality for the time between $t_0$ and $t_1$. It is a sequential pipeline, as individual assessment tasks are organized in a sequence. As soon as one step indicates that the view should not be considered further for composition, the remaining tasks within the pipeline need not be triggered (see Figure 45).

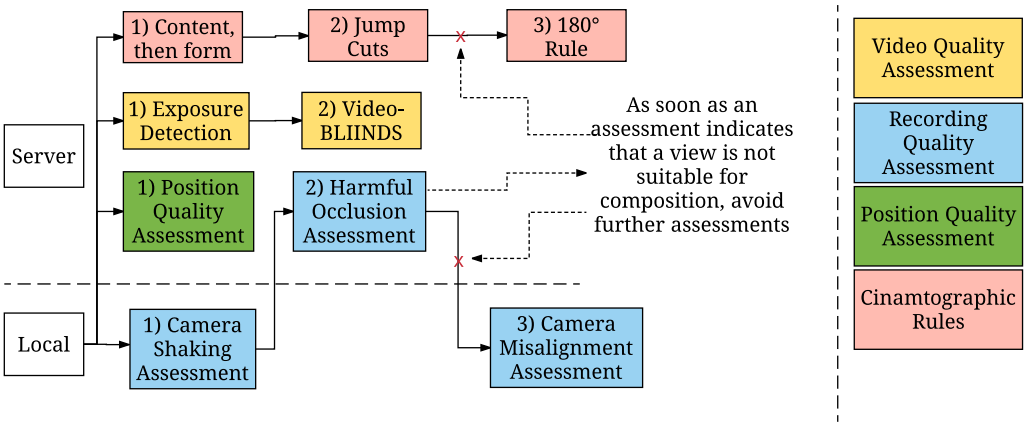



Figure 45: Filter Stage: Quality assessment pipeline for position quality, video quality and recording quality assessment as well as cinematographic rules.

### 6.2.1 *LiViU for Video Upload*

Before a quality assessment can be executed in the sequential pipelines shown in Figure 45, LiViU is used for the upload of video streams. The main task of LiViU in the context of video composition is to upload a stream to the video composition component, which is run on a central server. During the process of uploading a video stream, the filter stage of the video composition application is triggered, determining the views that will not be selected for composition. The filter stage of the video composition application leverages LiViU's capabilities to push or pull video streams.

### 6.2.2 *Quality Assessment using the PaSC*

The PaSC and related algorithms proposed in Chapter 4 are used for quality assessment in the categories recording quality assessment and video quality assessment. The reader should be aware, that the sequential pipeline of quality assessments is executed in a distributed manner which means that individual assessment steps are performed on different devices.

The sequential pipeline is constructed in a prioritized manner, with the most distracting degradations (i.e., in the recording quality assessment) analyzed first. The recording quality assessment algorithms discussed here were introduced in Chapter 4. The video quality assessment uses two complementary algorithms for under-/overexposition detection of video frames, as proposed by Saini et al. [**Saini2012**] and the V-BLIINDS algorithm [**Saad2012**].

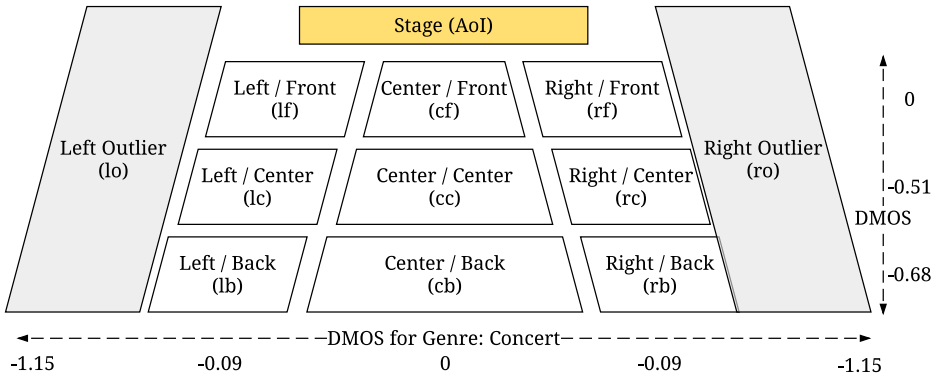



Figure 46: Mapping of recording location based on the recording distance and angle from event to a relative scene model, including the impact of the recording position on the perceived quality [DMOS].

### 6.2.3 *Recording Position Quality Assessment*

Recordings of the same scene differ regarding the distance to a scene and the recording angle. Chapter 3 proposed novel models that quantify the perceived quality in relation to its position.

Detection of a scene location is conducted using the novel AoI algorithms as proposed in Section 4.2.4.2. It leverages the compass readings from the recording smartphones and camera lense information to determine the FoV of all recording devices. On the basis of the FoV of many devices, the AoI can be determined. For determining the position of devices such as smartphones, a location provider is assumed, which offers reliable detection rates, such as GPS. To compensate for varying reliabilities of the location providers, the quality models are mapped into a relative scene model (depicted in Figure 46). It classifies the distances of recorders in the audience in relation to each other into close to the scene (f), central (c) or in the back (b); and the angle into left (l), right (r) and central (c). Zones are annotated with genre-specific quality values on the basis of the discussion in Chapter 3. This complete model gives no precise information on absolute positions and distances, but allows for a relative localization to cope with small imprecisions in smartphone sensors.

Indoor events require additional processing for retrieving the relative location of the recording devices. Systems that achieve a suitable detection of relative positions rely on Structure from Motion (SfM) and have been evaluated in the context of video composition by Arev et al. [**Arev2014**].

### 6.2.4 *Cinematographic Rules*

By the scene model discussed in the previous section, the cinematographic rules "content, then form," "180° rule," and "jump cuts" are validated. The camera misalignment algorithm as proposed in Section 4.2.4.2 is leveraged for scene localization. It relies on a majority consensus decision, which excludes recordings which do not capture the AoI. The agreement on an AoI allows to ensure that cinematographic rules are not broken for automatic video composition.

*"Jump cut"* detection relies on the zones determined in the scene model. As a result, when switching from one recorder to another, the new view shall not be captured within the same zone as the current recording. Also, it is combined with the "30° rule," which ensures that the relative angle between two recording positions should be at least 30°.

Similarly, for detecting if the recorded views comply with the "180° rule," the filter stage leverages the scene model and the orientation used by the built-in compass of the recording devices. On the basis of this information, those views are detected which have been captured from an opposite direction. These views are discarded from composition.

## 6.3 CROWDCOMPOSE

The set of filtered video views can either be composed by the semi-automatic composition engine CrowdCompose or the trained model of AutoCompose. CrowdCompose leverages the concept of crowdsourcing to recruit a large group of human workers, who complete predefined and small tasks to compose a video. These workers work in parallel on the small jobs and receive a monetary compensation for a successful completion.

The aim of CrowdCompose is two-fold:

1. To compose a video stream which achieves a perceived quality superior to existing automatic algorithms, and
2. to create composition models, which can later be used by the automatic composition algorithm AutoCompose.

### 6.3.1 *Architecture of CrowdCompose*

Figure 47 depicts the components required in CrowdCompose, which are distributed across several server instances to cope with a high number of concurrent users composing a video stream.

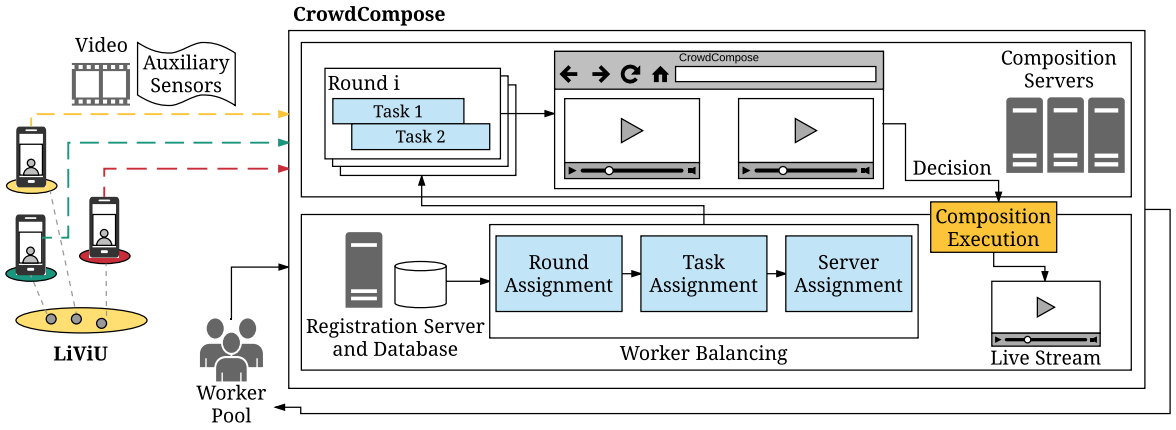



Figure 47: Architecture of CrowdCompose for conducting a semi-automatic video composition.

#### 6.3.1.1 *Overview on Different Servers*

CrowdCompose outsources the task of composing live video to a large workforce mediated by an online crowdsourcing platform, such as Microworkers[1]. It is assumed that this mediating platform offers a pool of workers which are instantly available to complete tasks. CrowdCompose assigns tasks to these workers and tries to keep them bound to the system as long as possible. The service consists of one central *registration server* as well as a scalable number of *composition servers*, which offer the composition UI. The infrastructure implies

1 www.microworkers.com; Visited on: 09/02/2016

that all servers are connected by high-speed networks, ensuring both low delay and an exchange of multiple parallel video streams. In the current setup of CrowdCompose, this is achieved by deploying all servers in a single data center. All servers share a dedicated storage on which incoming video views are stored.

*Registration Server*

The *registration server* is the initial point of contact for both the workers composing a video stream as well as LiViU when it starts streaming a video. For LiViU, the registration server mediates the recording device to the composition server, which receives the video stream. At any given moment, LiViU streams the video to only one server.

For both the workers and the recording devices, the registration server is used to ensure both a load and user balancing across the composition servers. Furthermore, the registration server is used to execute the video composition by stitching video streams to the composed video. Thus, the different composition servers submit the workers' decisions to the registration server.

*Composition Server and Clients*

The composition servers provide the web client UI. They receive the various synchronized video streams, segment them, and provide them to the clients in a single UI, which shows up to four video views of the same event in parallel (see Figure 48). The simplistic UI design was shown to be beneficial by Lasecki et al. [**Lasecki2011**]. Minimal interactions possibilities are offered to the user to allow workers to focus on a fast task completion. The respective tasks that are mediated to the workers are discussed in Section 6.3.2. Decisions made by the workers are transferred to the registration server to execute the video stitching.

As crowdsourcing implies mitigating streams to an anonymous crowd of people that could be located around the world, their Internet connection speeds may vary. A connection which does not allow one to stream multiple video views without stalling effects would reduce the speed of the composition and may lead to wrong decisions of workers. Each session is thus monitored to ensure that solely workers with sufficient network capacity use the system. This approach is discussed in Section 6.3.5.2.

#### 6.3.1.2 *Software and Libraries*

A single MySQL[2] database handles the storage of decisions and metadata of the streaming clients. All servers run a Web- (HTTP) and Java Application server[3] that enables running the web-based UI and allows inter-server communication. The web-based UI is implemented using standard Hypertext Markup Language (HTML) 5 and JavaScript.

Inter-server communication is achieved using Representational State Transfer (REST) web services, which encapsulate the public functionality of registration and composition servers. The registration server can access all videos in a shared storage space, which allows quick stitching into the composed video. The stitching of videos is achieved by using OpenCV 3[4], the open source framework for computer vision, and the video coding library FFMPEG[5].

2 http://www.mysql.com/; Visited on: 09/02/2016
3 http://tomcat.apache.org/; Visited on: 09/02/2016
4 http://opencv.org/; Visited on: 09/02/2016
5 https://ffmpeg.org/; Visited on: 09/02/2016

Similar to LiViU, it is assumed that NTP allows suitable synchronization of different video streams. The challenge of synchronizing media streams from different sources is intensively discussed in related work [**Guggenberger2015a**, **Guggenberger2015**, **Shrestha2007**, **Shrestha2010b**].

### 6.3.2 *Task Design*

CrowdCompose pursues the atomization and parallelization of the tasks to 1) select the next, best video view for a composed video, and 2) determine the point in time to switch to this new view.

A sequential pattern of the two tasks has been chosen in which Task 1 is generally responsible for agreeing on the next video view, whereas Task 2 is in charge of deciding when to switch views.

#### 6.3.2.1 *Task 1: Selection of the Best View*

Task 1 is split into two parts discussing the video and the audio track. Task 1a is assessing the video views and Task 1b the audio tracks. Task 1a and 1b rely on assessing based on the SSCQS for subjective quality assessment as discussed in Section 2.3.2. As a result, the MOS annotates each video view.

*Task 1a - Video View Selection*

*Task 1a - Video View Selection* asks workers to select the most suitable view. To reduce assessment times, the video stream is segmented into rounds of duration $t_r$ (round time). The selection is based on up to four synchronously played back video views for $t_r$ seconds. A reference view represents the media stream currently selected for composition. The reference view is used to normalize ratings across view groups in a round. Each view group contains at least two and up to four video views. The reference view is part of each view group. Our empirical study determined the maximum number of video views, as it allows users to retrieve the videos in an appropriate size and without the need to scroll[6]. Thus, at most four parallel views are clustered into view groups identified by an index. The video views are reduced in size to allow parallel viewing and timely decision making. The reference view is highlighted accordingly in the UI (see Figure 48). Assessment of different video views is possible during the playback of a video segment. The mean of the assessments of the workers is used to select the best view in each round and to annotate it with a quality value (MOS).

*Task 1b - Audio Track selection*

In *Task 1b - Audio Track Selection*, the appropriate audio stream is selected for the composed video. The aim of this approach is to ensure mostly noise-free audio in the composed video. Whereas the audio quality assessment algorithms aim at finding compression artifacts or technical degradations in a track, CrowdCompose aims to find noisy and clamorous recordings.

Only a subset of at most four audio streams is selected. The location in the scene model is used to determine those views recorded from a central position and that are at a close

6 For the design the mediating platform Microworkers was used which defines the UI size.

distance to the AoI (scene model: fr, fc or fb). Workers are asked to listen to the reference audio track and a single, new audio track - each of at most $\frac{t_r}{2}$ seconds.

In comparison to the video part - where diversity is intended - maintaining stability is important for audio. This has been shown to be beneficial for video composition by Wu et al. [**Wu2015**]. A switch is only invoked if the reference audio track achieves an MOS of less than 3.

#### 6.3.2.2 *Task 2: Timing of a View Transition*

*Task 2: Timing of a view transition* allows workers to give an answer to the question of when to switch from one view to another. The UI for Task 2 is illustrated in Figure 48, which includes the reference view on the left and on the right the best-rated video view. The worker selects the point in time when to switch from the left to the right video view.

*User Interface and Selection*

The timeline at the top shows both the time since task initiation and the decisions of other workers. Workers could jump back in time up to $b_c$ seconds and replay the video segment. Audio playback stems from the reference video view to determine a comprehensive overview of the best point in time to switch. Workers have the possibility to abort a switch from one view to another and thus indicate that the composition should play back the reference video view longer. If the reference video view is rated best in Task 1a, a switch to the second best-rated view is possible to ensure diverse compositions. For Task 2, the total time to find a suitable point for a shot transition is defined by $d_{max} = \overline{d_G} + 2 \times \sigma d_G$, where $d_G$ represents a video-genre-dependent threshold (see Section 6.5.3).

*Refinement of a Decision*

Workers may select very diverse switch points. This may indicate personal optima for placing a switch, where CrowdCompose needs a collaboratively agreed decision. To reach a common decision on when a switch shall be placed, a refinement approach is integrated into this task.

CrowdCompose monitors how many workers are currently working on Task 2 and the current time of assessment from a given, synchronized point in time when Task 2 started. This point in time continuously increases to $d_{max}$. Every second, it is evaluated if a window of 25% of the duration can be identified in which a majority of workers agreed on placing a switch. At least three workers have to place the switch point accordingly. As soon as this window is identified, the current round for Task 2 is completed, and no further decisions are accepted.

This design is a modified version of the crowdsourcing algorithm "rapid refinement" by Bernstein et al. [**Bernstein2011**]. In contrast to "rapid refinement", the proposed modification allows workers to see the selections of other workers on a timeline, with the ability to rewind playback and rethink a decision. The identified window is then used for a refinement of the task. Workers repeat the task for the previously determined video segment. The process is terminated either after two refinements or if a majority of all votes are within a three-second window. Video cut points are automatically determined by averaging all human decisions. No switch is conducted if the majority of workers decided not to switch.

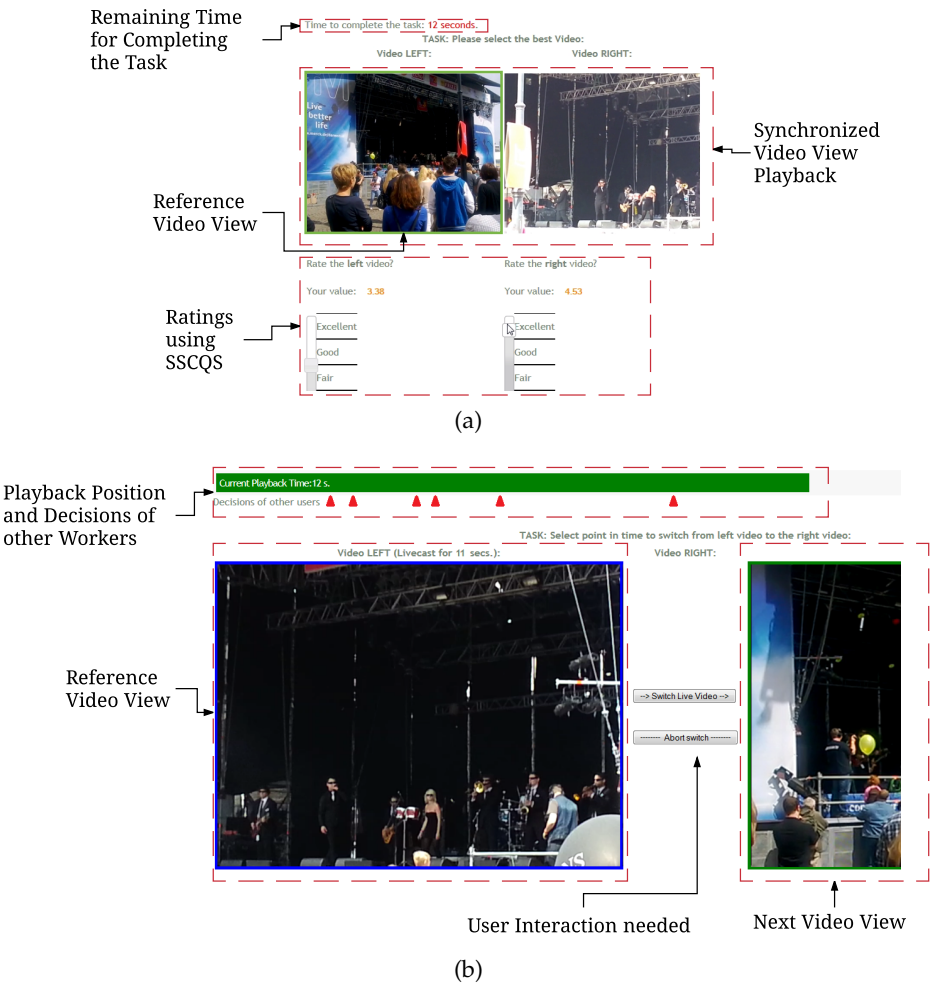



Figure 48: UI of the different CrowdCompose tasks: (a) UI of the CrowdCompose Task 1 - Assessing the quality of video views (b) UI of CrowdCompose Task 2 - Determining the suitable time for a switch in views.

### 6.3.3 *Round System and Playback Delay*

Timings and durations are important for understanding CrowdCompose, as the time workers need to make decisions delays the broadcast of a video. This delay is known as the broadcast delay and is calculated as $BD = n \times t_r + b_c + 5[s]$ .

For Task 1a the live stream is divided into segments, called rounds, of $t_r$ seconds of video. A number of $n$ rounds can be processed in parallel. $b_c$ describes the composition buffer in seconds which allows workers to rewind the video in Task 2 for placing a switch. Five seconds are reserved for stitching the composed video.

Increasing values of $t_r$ and $n$ delay the creation and playback of the composed video. Depending on the timing requirements, which can be specific to the video genre, both parameters $n$ and $t_r$ are adjusted (see Section 6.5.2). Thus, the parameters can be used to define thresholds that workers need to comply with. If they repeatedly break the given thresholds for executing the tasks, they can be excluded from using the system (see Section 6.3.5.2). Workers' ratings are discarded if they are not able to complete the task in time.

Figure 49 depicts the concepts of in-parallel processing of video streams. $n$ different rounds are depicted with each containing a fraction of the video stream. Five seconds

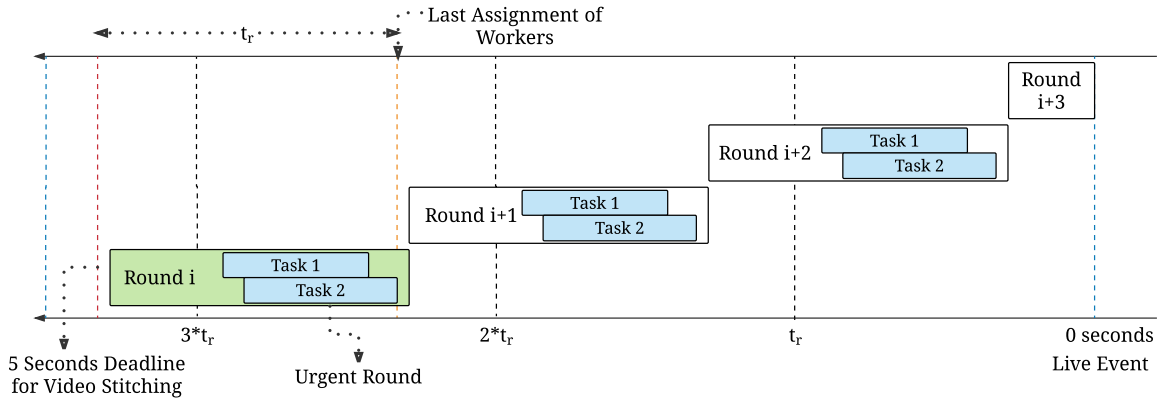



Figure 49: CrowdCompose's round system for parallelizing the video composition.

before the broadcast time, the decisions (Task 2) for a round have to be gathered, and the last assignment of workers is consequently possible $t_r + 5$ seconds before the live edge. BD can thus be adjusted to plan the delay between the reception of a frame on the servers and its distribution to the viewers of a video stream.

### 6.3.4 *Worker Balancing*

CrowdCompose integrates automatic balancing of workers across different tasks (see Figure 50). It pipelines workers to conduct the composition in parallel in two ways.

First, workers are split across different rounds and, second, across different tasks. By using this approach, a quick decision can be made when the number of workers is high enough. Task assignment prioritizes the evaluation of the video content quality in Task 1a. After receiving three valid assessments in Task 1a, the assignment of Task 2 is initiated.

If more than four parallel video views exist, the views are grouped into view groups, always containing the reference view and three more video streams. Each view group is assigned to a distinct CrowdCompose server to ensure an equal server load. Balancing checks upon the arrival of a worker if at least three ratings are gathered in the urgent round. The urgent round represents the segments for which no final switch decision is made, and which is closest to the live edge.

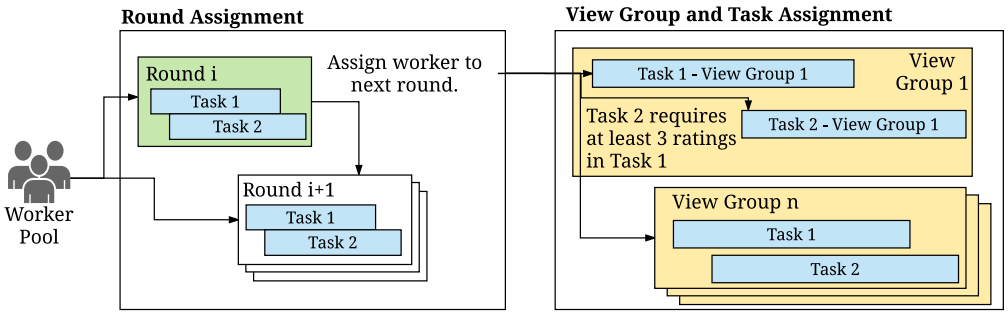
If this is ensured, the workers are assigned to the rounds in a round-robin manner. Groups of three workers are chosen as they allow at each time to retrieve biased assessment. To compare the ratings of different view groups, the ratings of the reference view are taken as anchor points.

### 6.3.5 *Challenges*

Two challenges can occur when using CrowdCompose: a varying number of available users in the worker pool and a varying reliability of the worker's decisions.

#### 6.3.5.1 *Varying Workforce*

Mediating crowdsourcing platforms are used to get access to the workforce. Depending on the time of day the number of workers which start composing a video may change - even within a video composition session.




**Minimum:** Three workers per task
**Prio. 1:** Ensure at least 3 workers in urgent round.
**Prio. 2:** Assign 3 workers to Task 2.
**Prio. 3:** Assign groups of 3 workers in round robin manner.

**Prio. 1:** Ensure enough workers in View Group 1.
**Prio. 2:** Assign 3 workers to each View Group in round robin manner.

Figure 50: Worker assignment strategy of CrowdCompose to different tasks and, if available, view groups.

*Queuing in Overcapacity Scenarios*

To compensate for this, the retainer model is used, which keeps workers in the system even though they are not assigned to a task [**Bernstein2011**]. When a video composition session is initiated, and workers are not immediately assigned to a task, the CrowdCompose website is kept open in a web browser to alert workers when a new task arrives. As long as the user is queued a minimal compensation is earned per minute - while no task has to be completed.

It is not intended to "over"-assess the views. Assessment in Task 1a and 1b is assigned up to M workers per round and task. M may range from 3 to 15 or even higher. If more than half of the assessors agree on the ranking of the best views, the assignment is stopped.

*Skipping Rounds in Undercapacity Scenarios*

If multiple view groups exist and no workforce is available, the completion of all tasks for one view group is favored over a partial processing of all view groups. Thus, a decision has to be made which views are selected for evaluation.

Based on the historic assessments of the previous round, the best views are selected. If they are dispersed across different servers, synchronization of the specified views is initiated to allow workers to make assessments on a single server. This is technically achieved as the server shares common storage space.

A switch to an automatic composition can be invoked if not enough workers are available to complete the composition.

### 6.3.5.2 *Reliability and Training of Workers*

The subjective assessments conducted in CrowdCompose require a reliable assessment from the majority of the workers. To ensure that both workers are well trained, and workers who cheat the system are detected, a qualification task is conducted that familiarizes each workers with the different task types. Gold standard questions, which ask workers about the content of the video stream, are used in this qualification task; this helps to detect workers who do not pay sufficient attention to the video streams. Only workers who achieve 100% correct answers are assigned to the worker pool of CrowdCompose.

Monitoring of the workers' response times and their assessments is essential in CrowdCompose. Regarding response times, too quick responses imply that the worker has not

been watching video views, whereas too high ones imply that the worker is not focused. Thus, after each round, a threshold is determined by $\overline{t_r} + 2 \times \sigma_{t_r}$ to potentially identify too slow answers.

If a worker breaks this threshold for two consecutive rounds, they receive a penalty. Penalties describe an increasing back-off time until the next task assignment for the misbehaving worker. After five back-off penalties, the worker is disallowed to contribute to the system. The basic principle of CrowdCompose assumes that the majority of workers perform well and that such a mechanism allows for detecting outliers.

As network speeds can highly vary during a streaming session, sessions are continuously monitored. Recurring stalling during the composition would slow down the decision process. Thus, workers are automatically withdrawn from a round if the stall duration is more than half a round ($\frac{t_r}{2}$ seconds).

## 6.4 AUTOCOMPOSE

AutoCompose takes the decisions generated by CrowdCompose and learns the composition patterns. Composition decisions are split into two separate steps, according to the pattern described in Section 6.1: first, into the selection of the next video view, and then the time when it should be integrated into the composed video. For automatic prediction, the machine learning algorithm SVM-HMM, a support vector machine combined with a hidden Markov model, is chosen.

### 6.4.1 *SVM-HMM*

As an efficient sequence tagging learning algorithm, an SVM-HMM is selected to understand the temporal sequence of shots in a composed video. The concept of SVM-HMM relies on the work of Altun et al. who show that it can outperform HMM for sequence prediction in both learning time and accuracy [**Altun2003**]. In recent years, further proposals have been made to increase the speed [**Tsochantaridis2004**, **Tsochantaridis2005**] and improve SVM internals such as a novel cutting plane approach for the SVM-HMM [**Joachims2009**].

Sequence tagging as video composition requires an analysis of the historic decisions to predict the next, best view and a suitable shot duration. The focus of AutoCompose lies on the video track, as it is assumed that the audio track shall only be switched if the audio quality suffers from major degradations [**Wu2015**]. This is ensured using the filter stage.

An SVM-HMM trains a model for an input sequence of feature vectors $x = (x_1, x_2, ..., x_n)$ which can predict the sequence of tags $y = (y_1, y_2, ..., y_n)$. The SVM is used for the formulation of the HMM. The trained model of an SVM-HMM is isomorphic to a k-th order HMM. A major advantage of an SVM-HMM in comparison to the classical HMM is that the observation variables $x_1, x_2, ..., x_n$ can be feature vectors and not only atomic values. This is leveraged as the input vector for the proposed composition algorithm consists of multiple input features.

A trained model predicts the next tag sequence $y$ as:

$$y = \operatorname{argmax}_y \sum_{i=1}^{n} [\sum_{j=1}^{k} (x_i \times w_{y_{i-j} \ldots y_i}) + \phi t(y_{i-j}, ..., y_i) \times w_t] \tag{29}$$

Here, $w_{y_{i-j} \ldots y_i}$ represents the emission vector, which is known from HMM and learned for each tag sequence $y_{i-j} ... y_i$. The transition vector, also known from HMM, is represented

by $w_t$ and gives the weights for the transitions between different tags. $\phi t(y_{i-j}, ..., y_i)$ represents a single value vector set to 1 to solve the sequence $y_{i-j}...y_i$.

### 6.4.2 *Features for AutoCompose*

Different features are used in an SVM-HMM to describe the video views. These characteristics are chosen as they influence how a video is composed. In detail, they influence the selection of the next view and its duration in the composed video. In an initial step, the view selection fuses the location of the current view, the genre of the video composition, and content characteristics. The video quality and the recording location determine the shot duration in the second step.

#### 6.4.2.1 *Location in the Scene Model*

Inputs for the SVM-HMM (see Figure 51) are the location and the orientation of a recorder classified in the scene model, as proposed in Figure 46 and Section 6.2.3. This represents a $3 \times 3$ field, including outlier regions with 11 states for the location. The value of the distance and angle are combined into atomic string values, e.g., a recording at the front right side as "fr".

#### 6.4.2.2 *Genre*

From professional compositions, it is known that the genre of a video has a major effect on the composition that is conducted. This affects the length a video view, as well as which video view shall be selected next. A limited set of genres are selected that are common in today's UGV: Sports (s), Music (m), Performing Arts (pa), Speech/Lecture/News (le), and other (o). The genres are represented as a single atomic genre string. The set of genres can be extended but that would require a retraining of the models. Each composition can be initially classified into a genre, e.g., by workers in CrowdCompose or by automatic genre classification algorithms [**Cricri2014**]. During a composition, the genre type may change, e.g., from music to speech.

#### 6.4.2.3 *Visual Features*

To train the composition model, the composed video as well as all available views from CrowdCompose, are analyzed using computer vision algorithms. These algorithms extract features and combine them with the data from the filter stage to learn composition patterns. Thus, the features assume that the content of the recorded scene and each view significantly affects the composition decision. The filter stage has ensured that a suitable quality is available in each view, and cinematographic rules are followed.

Visual features are chosen that describe video views for a certain duration. The ITU-T P.910 [**ITU-J2008**] and Hasler et al. [**Hasler2003**] have shown for video quality assessment that a classification of different video sequences is possible by describing the structure, motion, and colors. This motivates the selection of the SI for the description of structural information, TI for describing the motion, and the Color Perceptual Information (Co) for the number of colors. SI and TI, as proposed by ITU-T P.910 [**ITU-J2008**], and the colorfulness of selected video frames [**Hasler2003**] represent the content of a video by classifying each video view according to the amount of structures (edges), the motion and the different colors it captures.

SI applies a Sobel filter on each video frame $F_n$ from which the standard deviation of the pixel values is calculated when inspecting its luminance plane.

$$SI = max_t\{\sigma_{space}[Sobel(F_n)]\} \tag{30}$$

TI is calculated in a similar manner by considering the motion between to Sobel-filtered frames as $M_n(i,j) = F_n(i,j) - F_{n-1}(i,j)$. The indices $i$ and $j$ represent the row and the column of the frame pixel inspected.

$$TI = max_t\{\sigma_{space}[M_n(i,j)]\} \tag{31}$$

As a third criterion, the Co shows the perceptual differences in terms of colors in video sequences. It was proposed by Hasler et al. [**Hasler2003**]:

$$Co = \sqrt{\sigma_{rg}^2 + \sigma_{yb}^2} + 0.3\sqrt{\overline{rg}^2 + \overline{yb}^2} \tag{32}$$

The calculation is performed in the Red Green Blue (RGB) color space of a video frame. In the above formula, $rg$ is the difference between the red and the green channel of a frame ($rg = R - G$) and $yb$ subtracts the blue channel component to the other two channels: $yb = 0.5 \times (R + G) - B$.

The three metrics can be quickly calculated in parallel for a large number of views using either CPU or GPU support. The values of the metrics usually range between 0 to 100, and small deviations depict only imperceivable differences. In order to ease the learning phase of the SVM-HMM, features are rounded to the nearest multiple of 5.

The view is then selected based on the available highest quality that was measured in the filter stage. Furthermore, the filter stage excludes poor quality video views as well as recordings in conflict with cinematographic rules. In order to ensure diversity in the selection of the views, the composition algorithm switches to the second-highest quality view if the last composed view from this region is the highest quality view.

### 6.4.3 *Shot Duration*

For a selected view, the length of the segment is determined that is put into the output stream. The assumption when applying an SVM-HMM is that the positions and durations of previous shots affect the next shot duration. The duration of a shot is represented by integer values in seconds.

Features that allow the description are the location of the previous and current views in the scene model, as well as the genre of the current composition. The position of the current view is the result of the prediction of the recording position.

Also, the quality of the video view influences the duration. The filter stage ensures that all video views have a quality of at least of 3 or higher (MOS). To ease the training of the SVM-HMM, the quality determined in the filter stage is rounded to an MOS of 3 (sufficient quality), an MOS of 4 (good quality) and an MOS of 5 (high quality) before training. Saini et al. propose a similar idea [**Saini2012**]. In contrast to their approach, which adds a fixed bonus time on high quality video views, the proposed model learns the impact of quality on shot durations in relation to the recording position.

#### *Considerations When Using SVM-HMM*

The implementation of SVM-HMM[7] models and learns up to 400,000 different features, but suffers in training speed when non-binary features are used. The aforementioned features

7 https://www.cs.cornell.edu/people/tj/svm_light/svm_hmm.html; Visited on: 09/06/2016

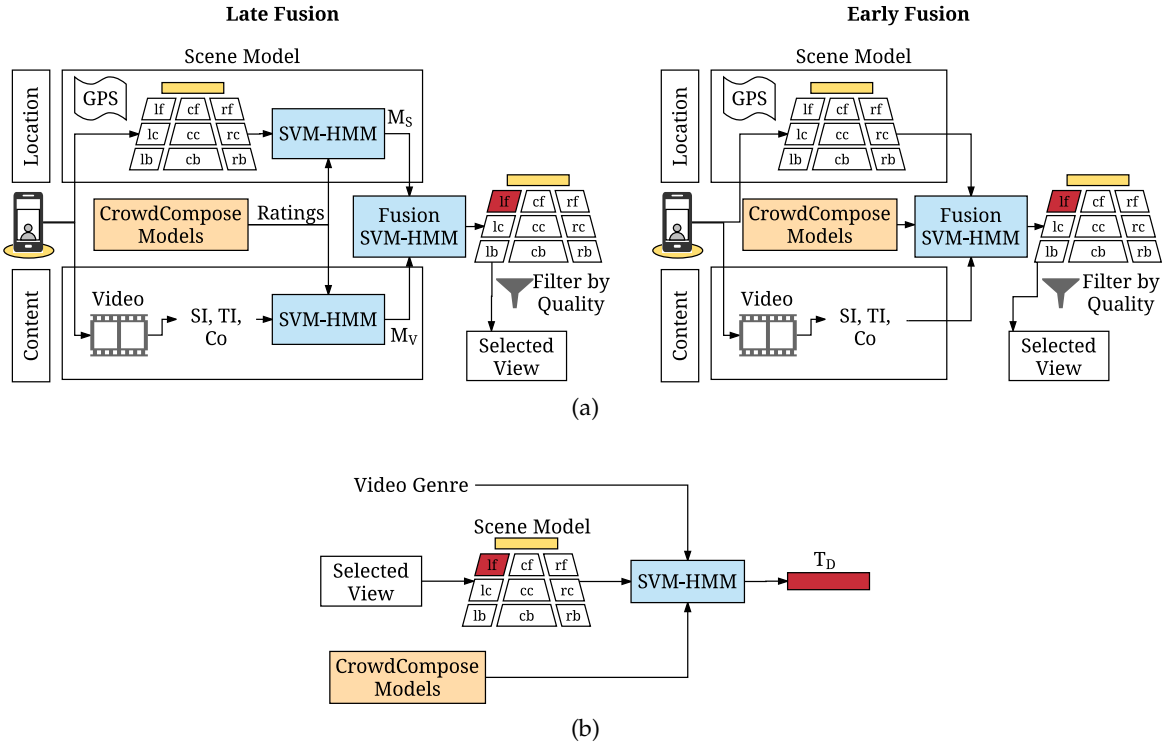



Figure 51: Concept of AutoCompose for automatic video composition: (a) Overview on the process of early fusion-based view selection in comparison with the late fusion for determining the next view and (b) Process to determine the video shot duration.

have been modeled in a manner to represent binary values. The quality feature as described in the previous subsection can originally be mapped to the set of $Q \in 3,4,5$. For training the SVM-HMM, it is translated into the three binary features: $Q_3$, $Q_4$, and $Q_5$.

### 6.4.4 *Learning the Video Composition*

Each composition decision of CrowdCompose is used to train the SVM-HMM. As input for the SVM-HMM, SI, TI, and Co are calculated for the respective video views and stored for each shot.

Also, the genre label of each sequence and the position of each video view in the scene model are derived from the location provider on the recording devices. These annotations are stored for the training for each composed video. Based on the values the classification step of the SVM-HMM is started and consecutively used to update the composition model.

When AutoCompose is used for composition, an once-trained model is used. For retrieving either the next view or the shot duration, a prediction relies on the same features for all video views that are received from the filter stage. Views need to be processed quickly for the video content characteristics and be annotated by the respective recording location and genre tag. The composition of AutoCompose, similar to CrowdCompose, allows a view switch every second. Thus, the characteristics SI, TI and Co are calculated for every second of each video view. The characteristics are continuously calculated and kept up-to-date.

### 6.4.5 *Early Fusion versus Late Fusion*

Figure 51 depicts different approaches for learning and predicting video compositions. The approaches are based on the concepts of early and late fusion of feature vectors [**Snoek2005**]. A fusion decision has to be made if features stem from different modalities. Early fusion combines the different features in a single feature vector representation and trains the SVM-HMM. This combination is achieved by concatenating the features [**Snoek2004**]. For the recording position selection, this would mean that the genre, content characteristics, and recording location are depicted in a single model. The trained SVM-HMM can then be used for predicting the next compositions.

The late fusion separates the features into their modalities, e.g., the location independent of video characteristics, and trains different SVM-HMMs. Thus, individual models first learn in an initial stage the semantics of the video composition, independent of each other. Probabilities or tag sequences retrieved from the first-stage SVM-HMM are then fused in a second stage SVM-HMM. The second stage gives the final result of the prediction. Early fusion is also described as the fusion of modalities in the feature space, whereas the late fusion is described as a fusion in the semantic space [**Snoek2005**].

In the conducted evaluations, late fusion has shown a negligible improvement of the correct classification rate for the JIKU video dataset [**Saini2013**]. The correct classification rate slightly decreased from 81.3% in the early fusion to 80.9% in the late fusion. At the same time, the learning and prediction time nearly doubled - making such an approach unfeasible for live video composition. The evaluated version of AutoCompose relies on the early fusion approach.

## 6.5 EVALUATING THE VIDEO COMPOSITION

This section describes the evaluation of the semi-automatic CrowdCompose and the automatic AutoCompose. Both approaches are discussed regarding the achieved quality of the video compositions, using subjective studies.

Finally, the performance of the contributions discussed in earlier chapters towards the video composition scenario is shown. The quality assessment using the PaSC and LiViU are discussed in Section 6.5.6.

### 6.5.1 *Experimental Setup*

The experiments performed for CrowdCompose and AutoCompose rely on the recording of five small- to mid-scale live events in 2014 in Singapore and Germany. Events E1 and E2 are used for evaluating parameter settings used in CrowdCompose. E3, E4, and E5 are the bases for evaluating the composition quality of both CrowdCompose and AutoCompose.

#### 6.5.1.1 *CrowdCompose Users*

Details of the users recording video and taking part in the composition on CrowdCompose are given in Table 17. Prepared recording devices using LiViU are used for streaming. Devices are either the LG Nexus 4, LG Nexus 5 or OnePlus One smartphones. Training was conducted right before the events, and included a description of the event and how to use LiViU. Users are allowed to select the recording position freely and to move as they desire. A common scene, e.g., the playing field of a soccer game, is given for orientation.

Table 17: Evaluation statistics for CrowdCompose and AutoCompose.

| | LiViU | | | CrowdCompose | | |
|---|---|---|---|---|---|---|
| | No. | Age | Male Users | No. | Age | Male Users |
| **Parameters** | | | | | | |
| E1-Sports | 4 | 20-28 | 4 | 46 | 16-42 | 36 |
| E2-City | 7 | 19-32 | 7 | 72 | 18-51 | 58 |
| **Evaluation** | | | | | | |
| E3-Music | 8 | 18-35 | 6 | 247 | 16-58 | 218 |
| E4-Sports | 12 | 20-32 | 10 | 277 | 17-55 | 233 |
| E5-News | 10 | 20-38 | 8 | 183 | 18-51 | 158 |

During the evaluation, up to 30 concurrent workers on each CrowdCompose server instance are permitted. Three composition servers and one registration server are used. In the evaluation section - unless stated otherwise - 10 minutes of the events E1 and E2 and 20 minutes of the events E3, E4, and E5 are evaluated. The statistical data for workers in CrowdCompose is given in Table 17. Additionally, the geographic region of the workers is included: 46.29% have their origin in Asia, 39.9% in Europe and 8.7% from North America. The remaining share is distributed over the rest of the world. Training of CrowdCompose workers is standardized by both a tutorial and a qualification task, as described in Section 6.3.5.2. Training included the description, testing of the tools, and the explanation of the payment scheme of ”punishments” and bonus.

#### 6.5.1.2 *Videos*

The recordings from event E1 last 12:23 minutes; for E2 13:45 minutes. Event E1 records different sports activities at the central university sports day from different perspectives. The event E2 is a tour in Darmstadt, Germany, showing PoIs and explanations by a guide. The evaluation video dataset includes recordings from a regional soccer stadium (26:59 minutes), a regional Music Festival (23:39 minutes) and the University Festival Report (22:33 minutes). The university festival report shows the annual celebration of the school. A speaker acts as a guide throughout the video, and diverse recordings are made - ranging from a stage including musicians to speeches by faculty members as well as an art exhibition.

#### 6.5.1.3 *Questions Discussed in this Evaluation*

Central questions in the evaluation of CrowdCompose are:

1. How can the parameters of CrowdCompose be set to allow workers make reliable assessments?
2. How much delay between the composition and the live event is needed for a good composition?
3. Which perceived quality does CrowdCompose achieve in comparison to automatic composition algorithms?
4. Can a sufficient worker pool be established to timely compose video?

From the composed video streams, AutoCompose learns how to compose a video. The central research question for AutoCompose is if the composed video streams achieve a superior quality in comparison with existing automatic composition algorithms.

### 6.5.2 *Parameter Study*

The first question for evaluating CrowdCompose addresses the optimal configuration to achieve a trade-off between the "liveness" of the composition and reliable assessments. CrowdCompose uses two parameters that need to be discussed: the length of a round $t_r$ and the number of parallel rounds $n$.

#### 6.5.2.1 *Broadcast Delay*

Both parameters have an influence on the broadcast delay. A high broadcast delay affects the composed video's perceived quality. Artificial delays are technically required and common in today's broadcasting environment.

To understand which delay is acceptable, an experiment with 33 assessors mediated over the crowdsourcing platform Microworkers[8] is set up. Different video genres are selected for the composition ranging from sports, news and music. A web page that includes a live newsfeed illustrating important events and a video being delayed by several seconds is shown. Users were asked to judge the impact of the delay on their viewing experience.

Sports broadcasts suffer immediately, as starting with delays of 20 seconds nearly the half of all viewers (43.1%), and for 40 seconds 68.4% of the viewers rated the delay as being distracting. Contrarily, for news broadcasts, more than half of the assessors accepted 30 seconds of delay or more.

The music performance was rated even less critical. For a majority of the assessors, a delay of 40 seconds was still acceptable. IPTV broadcasters such as Magine.TV[9] have a delay of 53 seconds for sports broadcasts[10]. CrowdCompose has to comply with the delay requirements. CrowdCompose's task design can be adjusted to comply with the findings and achieve a guaranteed broadcast delay of 20 to 40 seconds by setting the values for $n$ and $t_r$ accordingly.

#### 6.5.2.2 *Reliability of the Assessment*

The second research question addresses the reliability of the judgment of the workers. $t_r$ describes the round time, but additionally the time of the video segment which can be accessed by workers. It is thus a time limit for allowing workers to judge the quality of different views. Setting the value allows for complying with given subjective quality assessment rules (see ITU-R P.910) [**ITU-J2008**]. Videos shall be shown for around 10 seconds to allow a reliable assessment of the quality.

$t_r$ is evaluated for 5, 10 and 15 seconds. Table 18 shows the average required time for an assessor to complete Task 1a and the consistency on judgments for E1 and E2. The consistency of judgments defines the percentage of common judgments across all workers on the best video view.

For CrowdCompose a round time between 5 seconds and 10 seconds performs best. Whereas for E1 half of the workforce could complete the assessment in 6.8 seconds this took longer for E2 (8.3 seconds).

For E1 the consistency is highest at $t_r = 10$. In contrast to this, for E2 the system performs best at $t_r = 5$. Longer $t_r$ shows no improvement. Depending on the genre of the video, the round time is chosen with 5 or 10 seconds, respectively.

8 www.microworkers.com; Visited on: 09/02/2016
9 www.magine.tv; Visited on: 10/06/2016
10 www.heise.de/newsticker/meldung/Zahlen-bitte-Euro-2016-Manche-jubeln-erst-nach-56-Sekunden-3228756.html; Visited on: 09/25/2016

Table 18: CrowdCompose: Rating time and consistency of judgments across workers. Consistency of judgments depicts how many users agreed on the same view. The system was evaluated with two video views.

| | | $t_r$ | | |
|---|---|---|---|---|
| | | 5s | 10s | 15s |
| Rating time Median [s] | E1 | 6.8s | 13.1s | 14.9s |
| | E2 | 8.3s | 10.9s | 15.3s |
| Consistency [%] | E1 | 59% | 68% | 67% |
| | E2 | 63% | 54% | 56% |

Table 19: CrowdCompose: Fraction of agreements in a three-second window ($\geqslant$50% of the votes need to be in a window).

| M | | 3 | 6 | 9 |
|---|---|---|---|---|
| Initial Assessment | E1 | 7.9% | 10.2% | 3.9% |
| | E2 | 19% | 18.3% | 8.1% |
| Refinement 1 | E1 | 48% | 54% | 28.4% |
| | E2 | 31.1% | 27.4% | 24% |
| Refinement 2 | E1 | 39.6% | 26.8% | 38% |
| | E2 | 37.6% | 45.4% | 49.6% |
| No window found | E1 | 4.5% | 9% | 29.7% |
| | E2 | 12.3% | 8.9% | 18.3% |

The number of rounds $n$ determines how many video segments of a view are assessed in parallel. An increase of $n$ - the number of parallel rounds - enhances the rating consistency. The consistency increases by 11% until three parallel rounds.

With more than three rounds no increase in the consistency of judgments could be observed. The broadcast delay is chosen as a constant, independent value for each genre. Assessments of workers which require more than $n * t_r + b_c$ are discarded. The combination of $n * t_r$ is chosen to comply with $BD = n * t_r + b_c + 5$ being equal or less than the accepted delay for the genres music: $BD \leqslant 30$; sports: $BD \leqslant 20$ and news: $BD \leqslant 30$. It affects the overall broadcast delay. The feature was used in Task 2 at least once by 71.4% of all workers. If used, the average rewind time of workers is around 3.8 seconds. The parameter $b_c$ is set to 5 seconds for the system evaluation. It allows workers to rewind the playback by 5 seconds.

#### 6.5.2.3 *Assessing the Costs*

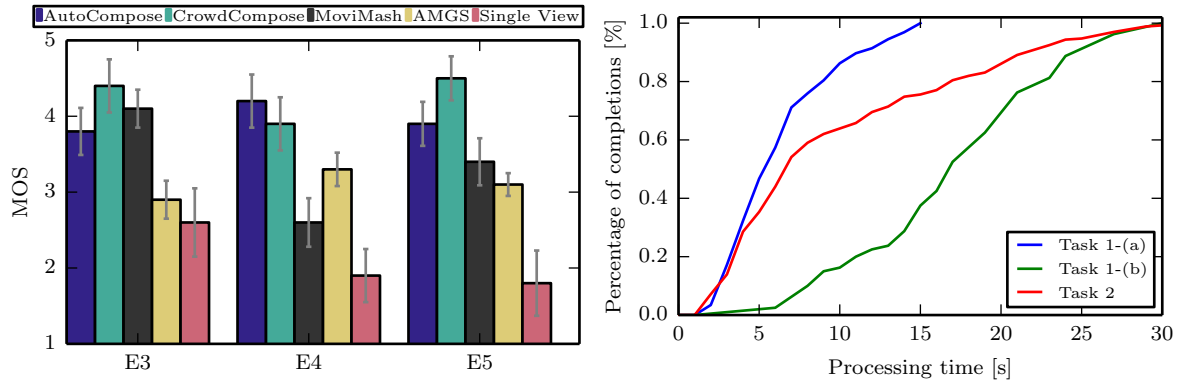
Workers cost money, so they should be wisely assigned to the individual tasks. The parameter M defines the maximum number of workers assigned to a task. M is a multiple of three. Table 19 shows the refinement tasks for view switches in Task 2 of CrowdCompose with different numbers of M per refinement for the events E1 and E2. It can be observed that the refinement steps allow a rapid agreement on a three-second window for determining a video view switch. Second, an increase of workers takes longer to agree to the window at least for $M > 6$. Additionally, Task 2 converges after two to three refinements, even with a low number of workers completing. M = 6 is chosen for the remaining evaluation to obtain reliable results in a short amount of time.

### 6.5.3 *Perceived Quality of the Composed Video*

Each of the composed videos is compared with automatic algorithms for composition AMGS, MoviMash and the presentation of the single, best video view [**Saini2012**, **Shrestha2010**].

The approach of Shrestha et al. optimizes the diversity of the shot selection and the overall quality using video quality assessment algorithms [**Shrestha2010**]. For improving the quality, it leverages an objective quality metric. In contrast to Shrestha et al.'s algorithm, MoviMash not only analyzes the video quality, but also introduces diversity regarding the recording position. It is thus the most similar algorithm to CrowdCompose. Both algorithms are described in Section 2.5.

Each event is assessed in a subjective quality study on the crowdsourcing platform Microworkers[11] by 35, 36 and 48 assessors[12]. From each composition, 60 seconds representing the same content are shown in random order. The assessors are asked to judge the composition on an SSCQS. The assessors do not know which composition algorithm created a video sequence.




(a) (b)

Figure 52: Evaluation results for CrowdCompose and AutoCompose: (a) Evaluation of the perceived quality for compositions of events E3-E5. (b) Cumulative Distribution Function on the response times of workers for a 10 minute segment in event E4.

The results of this assessment are shown in Figure 52 (a). It depicts that the human composition and the proposed CrowdCompose outperform an automatic composition using AMGS [**Shrestha2010**] and the presentation of a single view. The improvements can be achieved as the CrowdCompose composition achieves a stable video quality comparable with the one of the AMGS but places view switches better. The diversity of the CrowdCompose composition is higher, and shot durations are more stable in comparison to the automatic algorithms. Considering the third research question, CrowdCompose generates a high quality video composition which is superior to existing automatic composition algorithms.

The accuracy of the video switch placement (Task 2) is discussed to understand if the placement and resulting shot durations are suitable. A comparison with professionally produced content is made (see Table 20). Ten professional live TV broadcasts from the two public German TV stations ARD and ZDF are used. Shot lengths are determined in a manual annotation step. All recordings included video of more than 15 minutes and represented live footage of the genres "Entertainment," "News & Talks" and "Sports".

In Table 20 CrowdCompose achieves comparable shot durations but with a higher variance for news and music video. For professional sports broadcast the variance is higher compared to the composition of CrowdCompose, as, e.g., the minimum shot durations of 0.6 seconds cannot be achieved by CrowdCompose. The approach achieves the mean shot

11 www.microworkers.com; Visited on: 09/16/2016
12 Reliability check excluded the assessments of 9 (E3), 3 (E4) and 7 (E5) users.

duration of ”News & Talks” and ”Sports” live streams, but not of ”Music” videos. An explanation for the reduced average shot length in ”Music” live streams is that even though they are streamed live, most of the show is scripted. The director thus knows in advance when to switch to which camera view. In contrast to this, sports events have a higher uncertainty and thus can only be composed with a significant delay. Due to the three-second window in which the shot transition is determined, it is nearly impossible to achieve video shots of one-second length. An additional factor is that CrowdCompose generates outliers with long shot durations. Especially for recordings close to the stage (”fr,” ”fc,” and ”fl”) the manually determined shot durations are significantly higher than for those recorded at larger distances. This finding goes in line with the findings on the recording position discussed in Chapter 3.

Nevertheless, the median of 7 seconds (genre: sports) shows that CrowdCompose is not yet able to ensure a timely composition for sports. For the remaining genres, the round time of ten seconds is a good choice.

Table 20: Shot durations in professionally edited live streams in comparison to CrowdCompose and AutoCompose.

| | Professional | | | CrowdCompose | | | AutoCompose | | |
|---|---|---|---|---|---|---|---|---|---|
| | Mean | Median | Min | Mean | Median | Min | Mean | Median | Min |
| *Sports* | 11.4s | 6.4s | 0.6s | 8.4s | 7s | 2s | 11.1s | 6s | 3s |
| *News* | 17.9s | 13.9s | 1.1s | 13.6s | 12s | 3s | 15.9s | 8s | 3s |
| *Music* | 6.5s | 5.7s | 1.1s | 7.1s | 6s | 2s | 9.01s | 6s | 2s |

### 6.5.4 *Worker Task Times*

The response times for different task types in a real deployment of CrowdCompose are discussed now. For the event E4, a round time of five seconds at a total number of three rounds is chosen. As mentioned earlier, the broadcast delay is kept static over the complete broadcast. For the majority of workers, the successful completion of any task type is not a problem. Most workers completed Task 1a in less than seven seconds. The more time-intensive task of assessing the different audio tracks (Task 1b) is less time-critical. Only a small fraction ($< 25\%$) of workers require longer to complete. A majority of the workers complete the task successfully.

Task 2 also shows diverse response times. This is closely related to the refinement tasks. Workers of a first composition round watch the live stream and make their decision on a shot transition in a window of one second up to $\overline{D_G} + 2 \times \sigma D_G$. As the window length is reduced, refinement decisions must be given in shorter times as the window length is reduced.

### 6.5.5 *AutoCompose*

AutoCompose achieves a classification of a once trained composition model in real-time for the given models by leveraging content characteristics, a scene model and a filter stage that ensures cinematographic rules. It learns the composition styles for placing shot boundaries and selects views by composition results presented by CrowdCompose. The videos composed by AutoCompose are not part of the video set used to train the composition algorithm.

Events E3, E4 and E5 have been chosen to assess the performance of not only the CrowdCompose composition but also the AutoCompose composition. The resulting MOS is depicted in Figure 52 (a). It shows that the AutoCompose composition achieves a similar quality for events E3, E4, and E5 in comparison to CrowdCompose, and it outperforms MoviMash [**Saini2012**] in one of three events. At the same time and in contrast to MoviMash, AutoCompose conducts the composition in real-time. Also, MoviMash suffers from a quality decrease if the genre of the video changes. MoviMash was developed for music recording compositions as in E4, but suffers from reduced quality in the remaining genres.

As the differences are not significant for some of the events a second, independent forced choice evaluation is performed. Assessors are asked to compare two video sequences and to select the better one. The focus lies on the compositions by AutoCompose, CrowdCompose, and MoviMash. The sequences of 1) AutoCompose and CrowdCompose and 2) AutoCompose and MoviMash are compared. As a subjective quality metric, JND is chosen, as described in Section 2.3.2. The results show that CrowdCompose's composition is significantly better (JND $\geqslant$ 1) for event E3 and E5, and slightly better for E4 (JND=0.31). Except for the composition of E3, AutoCompose shows a significant improvement in perception compared to MoviMash (JND $\geqslant$ 1). Thus, it can be concluded that the perceived quality is different for most genres and superior to the presentation of a single video view without quantifying the difference. It is important to know that the results of forced choice experiments are not transitive; thus, it cannot be said that CrowdCompose achieves better results than MoviMash.

Furthermore, Table 20 depicts the resulting shot durations, which are in the case of AutoCompose different for each genre. This is intended for AutoCompose. AutoCompose's shot durations are slightly longer than those of the professional composition and the composition based on CrowdCompose.

### 6.5.6 *Supportive Applications*

During the setup of CrowdCompose and AutoCompose, the filter stage was used to eliminate poor quality video views. An essential part of this stage is the usage of the scalable quality assessment algorithms and the PaSC. Furthermore, the delivery of video streams is achieved by using LiViU.

#### 6.5.6.1 *PaSC within the Video Composition*

For the PaSC the video composition shows the real necessity for distributing quality assessment requests to mobile devices. As a result, the number of parallel video streams and their encoding parameters (resolution, frame rate and bit rate) is heterogeneous. The number of videos being processed lies between 4 and 12. The composition algorithm sets the deadline for processing requests.

In comparison to the synthetic work traces evaluated in the evaluation of the PaSC (see Section 4.4.2) the generated load on the devices and the server is lower. The potential for leveraging resources of the mobile devices is still huge. Using the devices for quality assessment nearly doubled system utilization (measured in CPU load) to around 28.3% in comparison to a single server setup. This allows the in-time processing of algorithms to be increased from around 68.8% (single server) to an average of 93.22%. For any video composition application, the usage of a PaSC system has the advantage of timely and more accurate quality assessments.

#### 6.5.6.2 *LiViU in the Video Composition*

LiViU achieves the upload of video streams. Due to its design principles, if videos do not pass the checks for quality and compliance to cinematographic rules, the composition server can significantly reduce generated data traffic. As soon as a video view breaks the constraints of the filter stage, the video stream does not have to be submitted. In the aforementioned events E3 to E5, this achieves an average data traffic reduction by 23.9%.

## 6.6 CONCLUSION

The proposed composition system consists of a filter stage, the semi-automatic CrowdCompose and the automatic AutoCompose. The proposed solutions comply with cinematographic rules and ensure a minimum video quality of the considered video views by using a filter stage in a first step. It achieves its real-time suitability by applying the concepts for video quality assessment as presented in Chapter 4.4.2 and allows a distributed execution of the algorithms. It leverages auxiliary sensor reading to construct a model of the scene, which ensures that basic cinematographic rules are not broken.

In a second stage, the composition is realized using CrowdCompose or AutoCompose. Both algorithms give answers to the central composition questions: "Which video view is selected?" and "When should a switch be realized?" CrowdCompose, a crowdsourcing-based algorithm, allows the near real-time composition by delegating the tasks of view selection and cut point placement to a group of people. Based on a majority consensus of the assessments, the best video view at any time and the ideal switching point are detected. To ensure a timely composition and to allow live streaming, the concepts of rapid refinement and pipelining of workers are introduced. The results for CrowdCompose show that a superior quality is achieved in comparison to automatic algorithms.

From the models generated with CrowdCompose, AutoCompose learns how to automatically construct a composed video in a quality-aware manner. It leverages an SVM-HMM to learn composition rules based on both content that is composed, and the location of recordings in a scene model. The composed videos lack quality in comparison to CrowdCompose, but are still preferred in comparison to other automatic composition algorithms. Video composition allows constructing video not only in a quality-aware, but also in an efficient manner. By using the LiViU, the generated data traffic is significantly reduced, allowing to furthermore reduce the uploaded data traffic under challenged network conditions and distribute solely one quality enhanced composed video.

# 7 CONTENT-AWARE VIDEO ADAPTATION

This chapter introduces the Video Adaptation Service (VAS), a support service for video streaming sessions, which considers both network and content characteristics of a video to improve the streaming quality and reduce the generated data traffic. By adding content-awareness to adaptive video streaming, VAS can achieve a better understanding on what is encoded in a streamed video, and which bit rate is necessary to achieve high quality for the user.

The contributions of VAS are two-fold. First, the system introduces adaptation support methods for content-aware video adaptation that are specifically designed for mobile devices. Current adaptation schemes are limited to a network-aware adaption - neglecting video content characteristics. VAS's adaptation schemes ensure a consistent quality level over long streaming sessions at lower bit rates than network-based adaptation schemes. They achieve this as in many situations higher bit rate representations do not offer perceived quality gains. It is shown that VAS is most beneficial for mobile streaming sessions which are executed in mobile networks. Cellular network users are usually bound to data-capped volume contracts, which allow them to access the Internet at high speeds for a limited amount of traffic per month. Users are interested in saving data traffic without sacrificing video quality.

Besides new adaptation schemes, VAS introduces a scheme that can easily categorize video content according to structural, temporal and color characteristics, enabling VAS to react quickly to changing content. VAS is suited for both VoD and live video streaming scenarios. The live streaming support is required for videos delivered by LiViU and video composition systems like CrowdCompose and AutoCompose. It is shown that this classification correlates well with subjective impressions of different video quality levels.

The description of the VAS revises our peer-reviewed publications [**Wilk2016c**, **Wilk2016b**, **Wilk2015**]. Also, we co-authored the quality assessment framework RT-VQM mentioned in this chapter [**Wichtlhuber2016**].

## 7.1 CONCEPT OF VAS

### 7.1.1 *Goals of VAS*

The vision for VAS is to enable high-quality streaming sessions with a minimum of generated data traffic. VAS is designed for HAS clients and focuses on mobile playback clients. In the remaining work, it is assumed that VAS leverages the video streaming protocol MPEG DASH. However, the proposed concepts can easily be mapped to other HAS protocols such as HLS [**hlsDraft**] or Adobe HTTP Dynamic Streaming[1]. Three subgoals are derived for VAS.

First, VAS aims to ensure a stable *perceived quality* during the entire video streaming session. Each streaming client can specify this desired perceived quality. VAS components are responsible for analyzing the video to determine the perceived quality of different representations and ensure that the video is played back at the desired quality level. If the

1 http://www.adobe.com/de/products/hds-dynamic-streaming.html; Visited on: 10/06/2016

content changes, e.g., after a shot switch, the perceived quality of a video representation may also change. VAS stabilizes the perceived quality and not the bit rate of a streamed video.

Stabilizing quality may introduce additional adaptations. The second goal of VAS is to perform adaptations between different video representations in an imperceptible manner. If adaptations between representations are performed too abruptly, the perceived quality may be additionally degraded [**Moorthy2012**, **Zink2003**].

Third, VAS aims at reducing data traffic by selecting the MPEG DASH representation that offers the desired perceived quality at a minimal bit rate. Users benefit from a reduction of monetary expenses in data-capped cellular networks. Data traffic in cellular networks is rather expensive compared to fixed network contracts, and it is often capped. For example an unlimited data traffic contract for LTE access costs approximately 159 Euro per month[2].

### 7.1.2 *Design Principles*

To achieve the aforementioned goals, VAS follows five design principles (DP):

DP1. VAS supports clients in adapting video. VAS is not executing adaptations on its own and complies with the MPEG DASH standard.

DP2. VAS requires only minimal client modifications to allow its quick adoption.

DP3. VAS recommends adaptations based on the perceived quality of the video and not the bit rate.

DP4. VAS uses image processing algorithms to estimate the perceived quality. This is a computationally intensive task that should not be performed on the mobile devices.

DP5. VAS's adaptation strategies should integrate existing research on video quality and adaptation effects on mobile devices.

VAS is designed to *support* video adaptation on mobile clients (DP1). Thus, the VAS server acts as a quality assessment proxy for a mobile device that wants to adaptively stream video. Videos are directly streamed from the video server to the client, as any additional streaming delay should be avoided - which would be introduced by content inspection and video quality estimation in the path. As a support service, one core principle of VAS is to allow MPEG DASH clients to request and execute adaptations on their own. Thus, VAS is not breaking the standard of MPEG DASH as it only offers an additional information source for making adaptation decisions [**Stockhammer2011**]. The MPEG DASH clients need a small modification that contacts VAS and integrates the returned data into their decision-making process (DP2).

Another reason for deploying VAS as a support service is the large number of calculations that it requires. The calculations are a result of VAS's perceived quality estimation of different video versions (DP3). The quality is objectively estimated with a high reliability using image processing algorithms (DP4).

The analysis of the perceived quality enhances existing adaptation services (DP5), which usually solely respect network conditions. Related research introduced new insights regarding how to execute adaptations (see Section 2.6.3). The adaptations introduced by VAS shall not degrade the viewing experience (DP5).

2 https://www.t-mobile.de/tarifoptionen/datenoptionen; Visited on: 06/09/2016.

## 7.2 THE ARCHITECTURE OF VAS

An architecture for VAS is derived from the design principles, as depicted in Figure 53. It shows the distinction between VAS servers and MPEG DASH clients which require minimal modifications to use VAS.

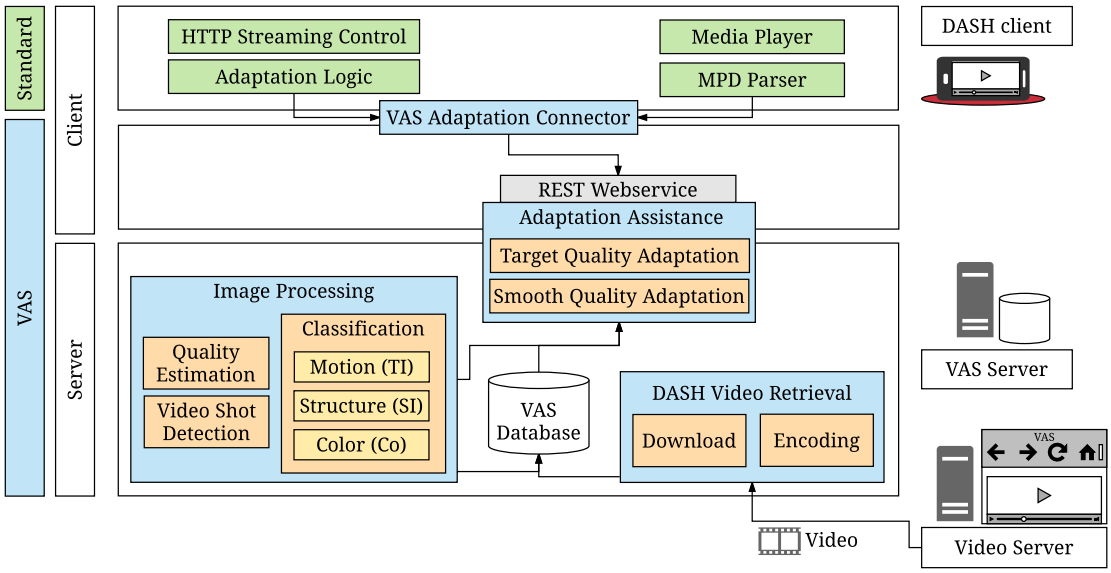



Figure 53: Components of VAS in relation to a MPEG DASH client.

### 7.2.1 *VAS Server*

The VAS server integrates the module for adaptation assistance and video retrieval, as well as the image processing-based video shot detection, quality calculation, and classification components.

#### 7.2.1.1 *Adaptation Assistance*

VAS is loosely coupled to the MPEG DASH client. To request the service's assistance, a MPEG DASH client accesses the VAS as a RESTful web service. This web service offers methods to

- initiate a new streaming session by providing the video's Uniform Resource Locator (URL) and display properties for retrieving a unique session identifier,
- request upcoming representations that offer the desired perceived quality by providing the session identifier and the desired quality; and
- request an adaptation plan for switching from a source MPEG DASH representation to a target MPEG DASH representation by providing the current application layer throughput and the buffer fill state.

With an initialization request of a streaming session, the client notifies the service about the client's available streaming resources and device characteristics - including its display resolution, decodable frame rate, and encoding support. The device properties determine the highest representation that VAS will recommend. Thus, the proposed service considers device specific requirements such as display resolutions, supported video encoding, and

the maximum decodable frame rate. Additionally, the client redirects the MPD URL to VAS.

The second method allows the client to determine which *perceived quality* it wants to stream. The Target Quality Adaptation (TQA) requires that for the whole streaming session, the perceived quality is known for each video representation.

Furthermore, TQA assumes that the throughput conditions are good for streaming the desired perceived quality. Based on the available network resources, the third request method allows retrieving an optimized adaptation plan when a client requires an increase or decrease of the video quality. VAS conducts these adaptations in a quality-aware and smooth manner using the Smooth Quality Adaptation (SQA). Both the TQA and the SQA are explained in detail in Section 7.4.

#### 7.2.1.2 *Video Retrieval and Pre-processing Stage*

As an independent service besides the video streaming server and client, VAS requires retrieving a copy of the video for analysis. The video copy is used to conduct a video shot detection and to estimate the perceived quality. If the video server and VAS are deployed in the same data center with a high-speed connection, downloading of all representations of a video is triggered. If this is not the case, or if VAS has to analyze the video of different content providers, only the highest available bit rate representation is transferred. This representation is used to re-encode all lower representations based on the parameters stored in the MPD.

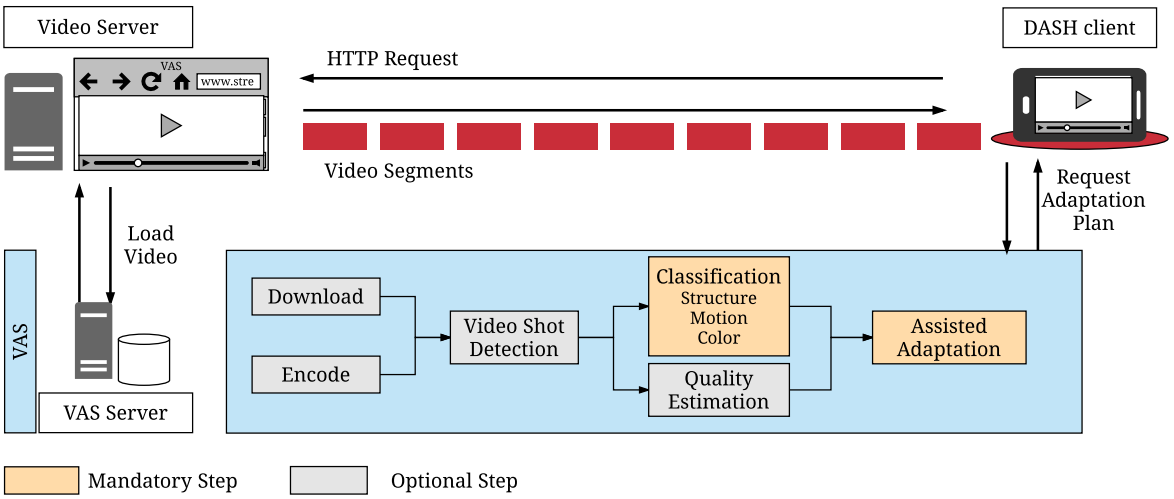



Figure 54: VAS preprocessing steps are necessary for classifying videos with similar content characteristics.

VAS offers adaptations in three quality dimensions of a video: frame rate, resolution, and SNR, which is solely influenced by the quantization parameter during the encoding. The SNR is approximated by the target bit rate of the video, whereas frame rate and resolution are usually described in the MPD, an MPEG DASH manifest.

The VAS analyzes the video which allows for adapting at runtime in a content-aware manner. To generate adaptation plans for mobile clients, VAS leverages a preprocessing stage that is triggered upon the first request of a video URL (see Figure 54).

As content characteristics may change over the course of a video, VAS determines chunks in a video stream in which the perceived quality level of a single MPEG DASH

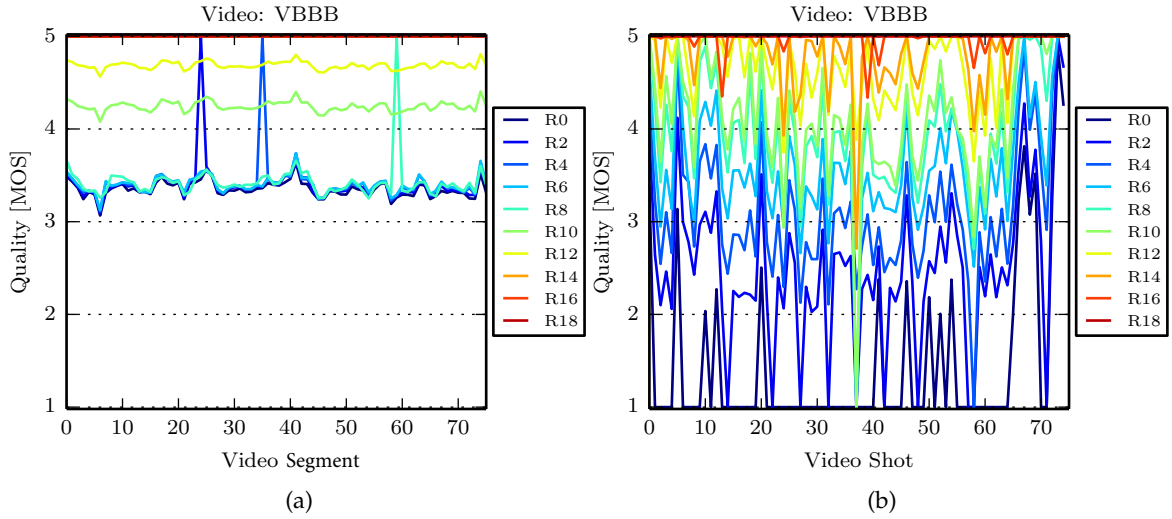
representation is constant. Figure 54 illustrates the preprocessing steps described in the following sections.

The depicted preprocessing steps show VAS's answers to the questions of a) What part of the video shall be analyzed? b) How can it be analyzed to quickly determine its perceived quality? and c) How can the process be improved regarding speed?

#### 7.2.1.3 *Chunk Preparation*

VAS estimates the perceived quality of different representations, but not for an entire video, as it is assumed that quality models may change over time for long-running videos. A quality model depicts the quality of a video chunk (e.g., regarding the MOS) for a given set of MPEG DASH representations. Thus, chunks of a video are prepared for which quality models are built. It was decided that video shots or MPEG DASH segments are chosen to determine these chunk boundaries. Whereas the MPEG DASH segment boundaries are described in the MPD, video shots are usually not signaled by streaming services. If MPEG DASH segments are chosen as a chunk for quality analysis, no additional preparation step is needed. Reliably creating quality models for video chunks is difficult for two reasons. First, perceived quality estimation using objective metrics (such as those proposed in Section 7.2.1.4) usually requires a minimal video duration. VQM requires a duration of more than three seconds which would not allow supporting MPEG DASH sessions with smaller segments. The second reason is the assumption that the perceived quality of a video changes with the content. Another disadvantage of MPEG DASH segments is the missing ability to map content-characteristic changes. Figure 55 compares the estimated quality models based on MPEG DASH segments with a duration of four seconds and based on video shots. MPEG DASH segments as video chunks have the disadvantage that the generated quality models are rather uniform over multiple segments.

In contrast, quality models change significantly over time for video shots. The perceived quality of different encoded representations of an MPEG DASH video shot is stable, whereas the perceived quality of different shots varies significantly. This is not a new phe-




(a) (b)

Figure 55: Comparison of the video quality models evaluated for the video sequence Big Bucks Bunny from the MPEG DASH dataset [**Lederer2012a**] using VQM. (a) Quality model calculated per four-second MPEG DASH segments and (b) Quality model calculated per video shots. Figures show every second representation.

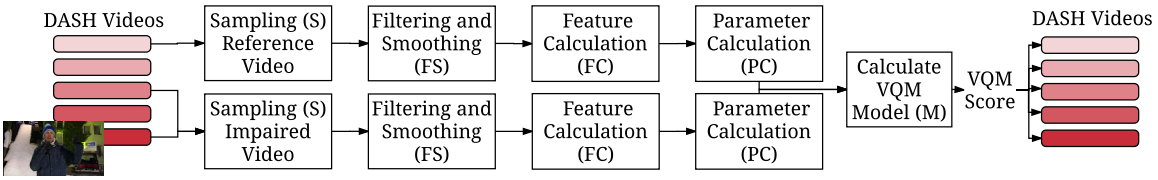



Figure 56: Essential steps of a VQM measurement between a reference representation and other MPEG DASH representations (inspired by Pinson et al. [**Pinson2004**]).

nomenon, but is supported for other multimedia applications by Adzic et al. and Akyol et al. [**Adzic2012**, **Akyol2007**]. The advantage of using video shots is that they quite reliably map content changes with the downside of an additional preprocessing time. This disadvantage is avoided if the video shots are already known from a composition service such as CrowdCompose or AutoCompose. If video shot boundary information is not available, VAS uses the edge-change ratio algorithm proposed by Zabih et al. [**zabih1999**]. The algorithm classifies the soon to-be analyzed segments with a hit rate of 86.95% for hard cuts [**Lienhart1999**]. In the remaining work, video shots are chosen for our quality models.

#### 7.2.1.4 *Quality Calculation*

VAS needs to understand the subjective perceived quality of each video representation over time. The service leverages an FR objective video quality metric, which can decode and analyze different MPEG DASH representations in parallel and build quality models for video chunks. An objective FR quality algorithm is used to determine the perceived quality of different video representations. Besides a reliable prediction of the perceived quality, a short algorithm runtime is favored to support live streaming or video conferencing scenarios. None of the existing algorithms combines reliable quality prediction and fast algorithm execution.

The VQM algorithm by Pinson et al. is selected to be used in VAS [**Pinson2004**]. The essential steps of VQM are depicted in Figure 56, and are described in Section 2.3.3.2. In contrast to the discussion in the background chapter, the quality of each MPEG DASH representation in relation to the highest bit rate representation is assessed. An algorithm execution starts with a selection of two different MPEG DASH representations: a reference and a representation to be assessed. The highest bit rate representation acts as a reference to determine the perceived quality of the lower bit rate representations. Remaining steps in this approach are similar to the previous discussion.

The quality assessment is repeated for each representation of a video, and quality values are calculated for individual video shots. The VQM values range from 0 (no differences between two video sequences to 1 (significantly impaired video sequence). For understanding what thresholds classify video into good and bad quality, a mapping is introduced to the well-researched MOS concept. This mapping was studied by Zinner et al. [**Zinner2010**].

The VQM implementation is not able to compare reference and test sequences in different resolutions and frame rates internally. A real-time quality assessment of VQM is developed which allows scaling of videos by leveraging the execution on a GPU. This version is called RT-VQM [**Wichtlhuber2016**], but is not part of the discussion in this thesis.

#### 7.2.1.5 *Classification of Video*

As the video quality calculation is computationally intensive, VAS classifies the content of a video by visual features and links them to quality models. Three content characteristics

are used for classifying video chunks: colorfulness, structural intensity, and the amount of motion. By classifying video chunks, quality models for one video chunk can easily be mapped to any chunk with similar characteristics - even for different videos. A detailed explanation of the contribution of the classification of video chunks is given in Section 7.3.

### 7.2.2 *VAS-enabled MPEG DASH Clients*

MPEG DASH clients that want to use VAS must allow a minor modification of the video adaptation module. It is assumed that the MPEG DASH client stores streamed segments in a playback buffer to compensate for unstable delivery times of video segments. The filling of the buffer and the adaptation logic of the player are not affected by using VAS; however, they can be supported.

Upon starting a streaming session, the client redirects the MPD location to the VAS server. This triggers the VAS server, which starts evaluation of the video stream to support adaptation. The mobile device decides when to consult the service. For example, the device contacts VAS in cases of throughput fluctuations that force a client to adapt. In any case, an initialization request is sent to the VAS server, triggering the video classification and preprocessing.

For assistance during the streaming session, MPEG DASH clients regularly request assistance by the VAS via a RESTful web service. The web service offers methods (as explained in Section 7.2.1.1) for recommending adaptations to reach the desired quality, and the imperceivable adaptation between representations. The remote VAS server is used for calculating content-aware adaptation plans and returning them to the MPEG DASH client.

## 7.3 CHARACTERIZATION OF VIDEO CONTENT

### 7.3.1 *Idea of the Categorization*

The main contribution proposed in this section is the reuse of existing quality models generated for a specific video for other videos. To understand which existing quality model should be used for an unknown video, VAS classifies videos. It is assumed that videos in the same class have similar quality models, and thus require similar quality adaptations. A classification of the video characteristics simplifies quality estimation.

The classes are characterized by visual features that represent the video content. To reduce VAS's operational costs, these features should be much easier and faster to produce in comparison to the execution of VQM. To support live video streaming scenarios, a real-time calculation of the features is necessary.

### 7.3.2 *Features for Classification*

We select features which are inspired by existing knowledge of the human visual system. As mentioned above, the features that are selected represent the structures in a video frame (SI), the motion between video frames (TI) and the colorfulness (Co) of a video [**ITU-J2008**, **Winkler2012**]. These characteristics, which are calculated for each video chunk, are analyzed as they indicate how the human vision system perceives video quality [**Winkler2012**]. The ITU recommendation has shown that specific metrics are reliable for classifying short video sequences in the context of subjective video estimations. VAS leverages the recommendations of the ITU, which are introduced for the video composition application in

Section 6.4 for SI, TI, and Co [**ITU-J2008**]. These metrics are calculated for individual video shots. VAS leverages the metrics to select appropriate quality models from the VAS database. It stores for each video shot the content characteristics along with the generated quality models. An advantage of SI, TI, and the quality estimator VQM is that they share a Sobel filter calculation to detect edges. Thus, a Sobel-filtered video frame can be reused in multiple processing steps.

### 7.3.3 *Selection of Characteristics*

Based on this classification, suitable adaptation plans are generated as VAS assumes that perceived quality differences of video representations encoded with the same resolution, frame rate, and bit rate are similar if the SI, TI, and Co values of the video shots are similar. The calculated adaptation plans, available video representations, and SI, TI, and Co profiles of a video are stored in a database of VAS (see Figure 53). The stored combinations are used to ensure the timely calculation of adaptation plans. As it cannot be guaranteed that video quality estimation of different representations can be achieved in time in any situation, previous quality estimations and their SI, TI, and Co values are stored in the database.

Unknown video sequences can be classified solely based on retrieving the SI, TI, and Co profiles. The profiles are then compared via nearest-neighbor matching (i.e., Euclidian distance of SI, TI and Co values) with existing profiles from the VAS database. To select an appropriate video shot from the database, the closest match is determined by the Euclidian distance:

$$\mathrm{ED}_{\mathrm{SI,TI,Co}} = \sqrt{(\mathrm{SI}_{\mathrm{DB}} - \mathrm{SI}_{\mathrm{Video}})^2 + (\mathrm{TI}_{\mathrm{DB}} - \mathrm{TI}_{\mathrm{Video}})^2 + (\mathrm{Co}_{\mathrm{DB}} - \mathrm{Co}_{\mathrm{Video}})^2} \tag{33}$$

In this equation, $\mathrm{SI}_{\mathrm{DB}}$ represents the SI value for a video shot stored in the VAS database, whereas $\mathrm{SI}_{\mathrm{Video}}$ represents the respective value of the currently played-back video stream. The aim is to minimize $\mathrm{ED}_{\mathrm{SI,TI,Co}}$ and to choose the reference with the smallest difference to predict the adaptation behavior. The adaptation plans of the match in the database are applied to the new video sequence.

The Euclidian distance weighs all features (SI, TI and Co) equally and does not prioritize any feature. Winkler et al. report that no clear preference to a feature can be given when classifying video content [**Winkler2012**].

This is the basis for selecting an appropriate adaptation plan based on the perceived quality, as shown in Subsection 7.4. In the preprocessing stage, the SI, TI, and Co calculations can be massively parallelized across different video shots and different representation combinations.

### 7.3.4 *Video Characteristics and Quality Models*

The assumption is that MPEG DASH videos showing similar content have alike quality models, and thus benefit from the same adaptation plans.

#### 7.3.4.1 *Quality Model Prediction Error*

The assumption that VAS can use existing quality models from videos with similar content characteristics is validated by investigating the average error when conducting such a mapping. For a video shot, the average error regarding the DMOS shows the average, absolute distance between another shot's quality model with a similar SI, TI, and Co profile

and the MOS predicted by VQM. The similarity between the video shots is measured using the Euclidian distance, as described for VAS in Section 7.3.3. The videos from the MPEG DASH dataset - provided by University of Klagenfurt, Austria - are used for validating the assumption [**Lederer2012a**, **Lederer2013**]. The dataset contains seven long-running videos encoded in 17 to 19 representations from 320x240 (240p) to 1080p resolution. All MPEG DASH representations of a video have the same frame rate. The characteristics of this MPEG DASH dataset are extensively discussed in Section 7.5.1.1.

The complete preprocessing stage is applied to all representations of all videos in the dataset to retrieve the SI, TI, and Co profiles and quality models for all video shots. Figure 57 depicts the average error when predicting a quality model for a video shot in relation to the Euclidian distance of both the predicted and the real quality model.

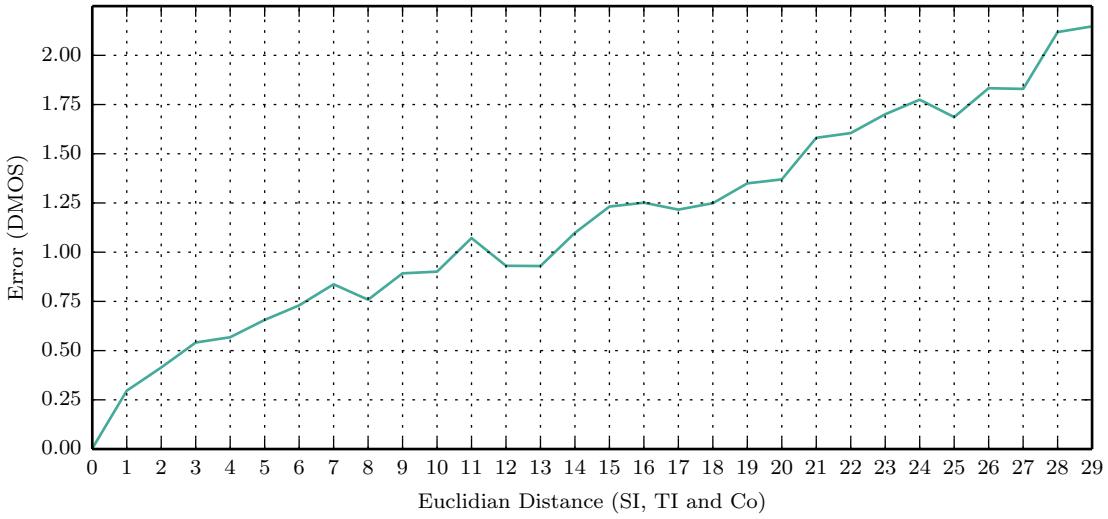



Figure 57: Average error for the prediction of a quality model with similar content characteristics.

If a shot can be found with exactly the same content characteristics, the quality model fits best. In this case, the average error is close to zero and below any noticeable difference for a human observer. With increasing distance, the error also increases. This shows that a single quality model is not suitable for all video shots. Up to an Euclidian distance of approximately 10.5, the average error stays below a DMOS of 1. Still, VAS needs to find similarly encoded video shots, where the average distance of content characteristics is as small as possible. Euclidian distances below 3 lead to only negligible deviations from the correct quality model.

#### 7.3.4.2 *Influence of Video Dimensions on Reliability*

Content characterization is further beneficial if the necessary calculations are made as quickly as possible. One possibility to reduce processing time is to limit the maximum resolution, bit rate and frame rate of the video.

The SI, TI, and Co profiles can be calculated not only from the highest representation but also from any other. The question arises: how inaccurate are calculations when a lower representation is chosen? Using the ITEC MPEG DASH dataset from Klagenfurt, the error in predicting the SI, TI, and Co values is calculated. The reference is the highest available representation, which is not depicted in the graphs. The representations are sorted by an increasing resolution and bit rate, meaning, that two representations with similar resolu-

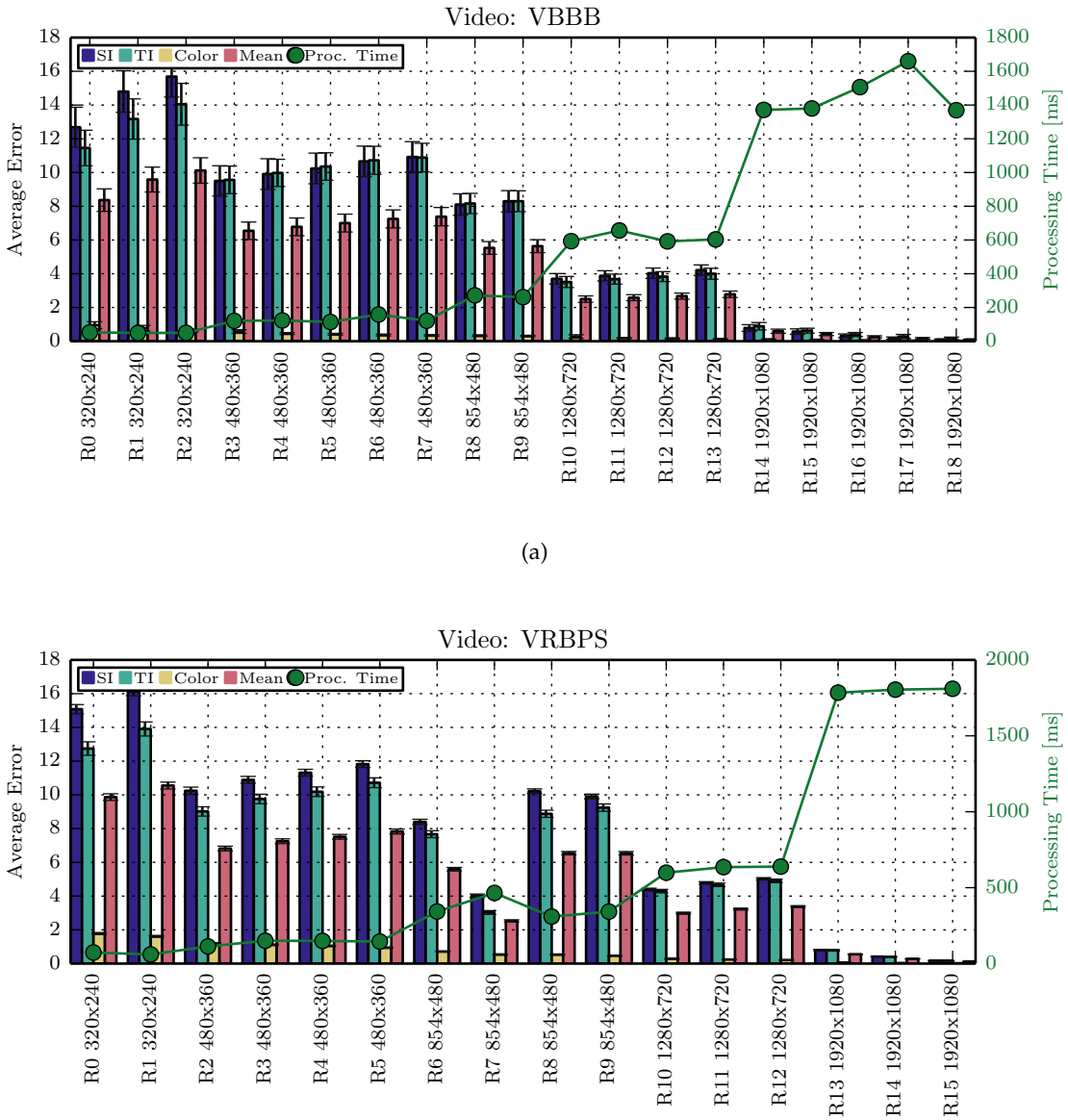



Figure 58: Average error for predicting the SI, TI and Co characteristics at low bit rate MPEG DASH representations for the (a) VBBB and (b) VRBPS videos.

tions differ regarding the quantization used for encoding. The frame rate is held stable as a decrease in frame rate results in a high average error for predicting the video's TI characteristics. Results are depicted in Figure 58 for the videos VBBB and VRBPS, which can be mapped to all other videos in the dataset. The processing time is calculated based on the sequential processing of the SI, TI, and Co on a single dedicated CPU core of an Intel Xeon CPU E5-1650 v2 @ 3.50GHz for one second of video.

The results illustrate that an increase in the resolution significantly reduces the prediction error but a linear relation between the resolution and the processing time exists. The quantization or bit rate has no significant (95% confidence intervals) impact on the processing time, but it slightly reduces the average error (red bar) as more details can be encoded. The increased processing time is a result of the data increase that needs to be loaded from the hard disk to the memory. A resolution of 720p for a 1080p reference video, or

480x360 (360p) for a 576p reference video, sufficiently shows both low prediction errors and more than half on the required processing time. These findings coincide with the results on the reliability of extracting other visual features from lower resolution videos as described by Manweiler et al. [**Manweiler2012**].

It can be concluded that a classification of MPEG DASH videos can be quickly and reliably done for low resolutions, and a complete quality model can be mapped by only inspecting a single MPEG DASH representation.

## 7.4 ADAPTATION STRATEGIES

The previous sections describe the architecture of VAS and the concepts necessary for quality- and content-aware adaptation for video streaming. This section describes the proposed VAS adaptation schemes, beginning with an optimal adaptation model that relies on global knowledge. The proposed MILP model guarantees stable quality with minimal data traffic.

Also, two adaptation heuristics are realized, which help MPEG DASH clients to adapt: 1) TQA and 2) the SQA. The TQA implies that a MPEG DASH client specifies a minimum level of quality regarding the MOS for the streaming session. VAS calculates an adaptation plan over different video shots, maintaining exactly the specified level of quality with only minimal fluctuations. This adaptation plan can be device-specific, i.e., including a certain maximum resolution, frame rate, or a specific set of encodings. To adapt across multiple MPEG DASH representations, the Smooth Quality Adaptation (SQA) adapts as covertly as possible. SQA is introduced in Section 7.4.2.2.

### 7.4.1 *Optimal Adaptation*

To determine what is potentially possible, the proposed adaptation heuristics are compared with an optimal adaptation scheme having global knowledge. An optimal adaptation scheme is formulated for a single playback device, in the form of an MILP that extends the work of Hossfeld et al. [**Hossfeld2015a**]. In comparison to their work, we focus on a model for MPEG DASH and we do not leverage the advantages of MVLC. Furthermore, our aim of the proposed model is very different, as it stabilizes the perceived quality level and minimizes the data traffic generated. It aims at streaming the lowest bit rate representation, which provides the maximum quality.

As a result, a two-step optimization approach finds a solution that achieves a video adaptation aimed at a target quality level (MOS) with minimal switching. The aim of the first optimization is to maximize the streamed video quality $W$ for a streaming session represented by the MPEG DASH segments that are received at the client ($i = 1..n$) and for the representations of the MPEG DASH video ($j = 1..r_{max}$). The objective to maximize the quality of a streaming session $W$ is formulated as:

$$max(W = \sum_{i=1}^{n} \sum_{j=1}^{r_{max}} w_{ij} * x_{ij}) \qquad where \quad x_{ij} \in \{0, 1\} \tag{34}$$

The weight $w_{ij}$ represents the MOS value of the representation $j$ and segment $i$. In this optimization model, it is assumed that the representations for a segment index $i$ are in an ascending order with respect to their perceived quality (MOS).

In additions, conditions are applied to the optimization model:

$$\text{subject} \quad \text{to} : \sum_{j=1}^{r_{max}} x_{ij} = 1 \qquad \forall i = 1, .., n \tag{35}$$

$$\sum_{i=1}^{k} \sum_{j=1}^{r_{max}} S_{ij} * x_{ij} \leqslant V(D_k) \qquad \forall k = 1, .., n \tag{36}$$

Here, $x_{ij}$ represents a binary value indicating if a representation $j$ is streamed for a segment $i$ or not. The first condition ensures that only one representation is downloaded per segment (see Equation 35). Equation 36 illustrates the sum of streamed data. Here, $S_{ij}$ describes the data traffic generated when streaming a segment at index $i$ and representation $j$. Note, $S_{ij}$ can be simplified to $S_j$ if a constant bit rate per representation is assumed. By using the segment $k$, we ensure that only the amount of data can be streamed that is available up to segment $k$. This amount of data is determined by a data trace and described as $V(D_k)$.

$$w_{ij-1} * x_{ij} < w_{ij} * x_{ij} \qquad \forall i = 1, .., n \quad \text{and} \quad \forall j = 2, .., r_{max} \tag{37}$$

Finally, equation 37 ensures that the minimum bit rate representation for a given MOS is selected.

With the given quality $W$ the second optimization step is executed. The aim is to stream video with a minimal data traffic for a given quality and a minimal number of switches. The optimization goal is set as

$$min(\frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=1}^{r_{max}} (x_{ij} - x_{i+1j})^2) \qquad where \quad x_{ij} \in \{0, 1\} \tag{38}$$

This equation implies that the number of switches for a given MOS is kept minimal. In addition to Equations 35, 35, 36, and 37, one additional condition is

$$\sum_{i=1}^{n} \sum_{j=1}^{r_{max}} w_{ij} * x_{ij} \geqslant W \tag{39}$$

This formula ensures that the chosen representations in the second optimization step are of at least the quality determined in Step 1.

The optimal model is evaluated in the evaluation section together with the adaptation heuristics explained next.

### 7.4.2 *Heuristics for Quality Adaptation*

#### 7.4.2.1 *Target Quality Adaptation (TQA)*

In comparison to classical bit rate-based adaptation methods (see Figure 59 - left), VAS assumes that the perceived quality does not necessarily increase with an increase of the bit rate. The TQA acts similarly to the optimal adaptation scheme but relies solely on local knowledge of the current and past throughput rates and perceived qualities of the adaptation.

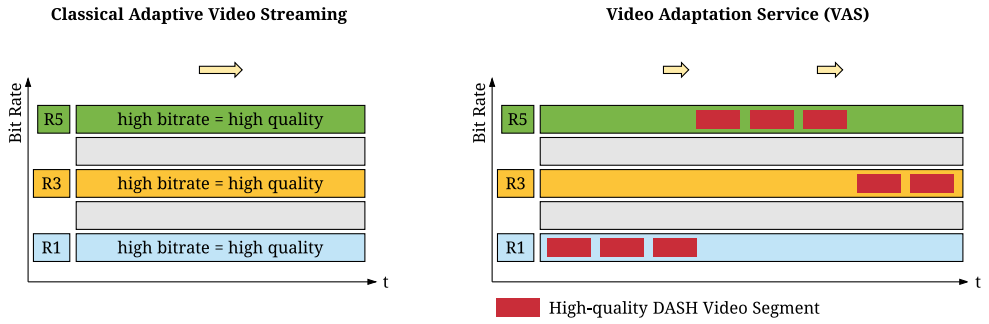



Figure 59: TQA in contrast to classical bit rate-based adaptation logics.

The TQA aims at serving an MPEG DASH client exactly at the desired perceived quality level independent of the bit rate. An MPEG DASH client can specify the target quality in terms of the MOS during the initialization request, which triggers VAS to analyze a video. Over the remaining course of a video, the strategy tries to keep this perceived quality level - which may result in switching representations even if the throughput conditions are stable (see Figure 59 - right). Thus, the proposed scheme may introduce additional switches. The chosen representations are dependent on the streamed video's content characteristics. In video shots with little to no motion, even representations encoded at low frame rates such as 6 FPS may be sufficient to achieve a high perceptual quality; whereas within the upcoming video shot, the motion may increase which requires one to scale the frame rate up to the highest possible representation. Similarly, the amount of structures and colors in the video shot influences the need for high resolutions to display the video in high quality. Ideally, those adaptations are placed at the shot boundaries identified by VAS. Note that if the shot boundary is not equal to the end of an MPEG DASH segment, adapting at a shot boundary can usually be achieved solely by an MPEG DASH client that supports HTTP GET requests for byte ranges (partial HTTP GET requests).

#### 7.4.2.2 *Smooth Quality Adaptation (SQA)*

Adaptations in a video stream can significantly impact the perceived quality, especially if an adaptation reduces the bit rate. Intensive research has been conducted on the impact of adaptations on the perceived quality (see the discussion in Section 2.6.3.2). Ideally, the adaptations decreasing the quality should be planned to be as covert as possible. These decreasing adaptations are executed when the end-to-end throughput decreases and thus the current quality representation cannot be streamed anymore.

At the beginning of a smooth adaptation, the MPEG DASH client informs VAS of both the currently played back representation and the target representation it needs to adapt to. The core idea of the scheme is to *slowly* adapt towards the target representation by integrating intermediate adaptations which are nearly unnoticeable. VAS relies on the findings of subjective studies of Moorthy et al., which offer thresholds on the perception of quality differences on mobile device displays [**Moorthy2012**]. Also, the playback buffer is leveraged to delay adaptation as much as possible, to keep the streaming experience high without risking stalling effects. The playback of video streaming at the highest quality representation has been shown to be beneficial [**Hossfeld2014**, **Moorthy2012**]. It leverages both the available buffer and the current throughput to determine the point when the complete adaptation must be completed. To reduce the risk of stalling, the threshold $B_c$ is defined, which limits the amount of buffered data to be used for delaying the adaptation.

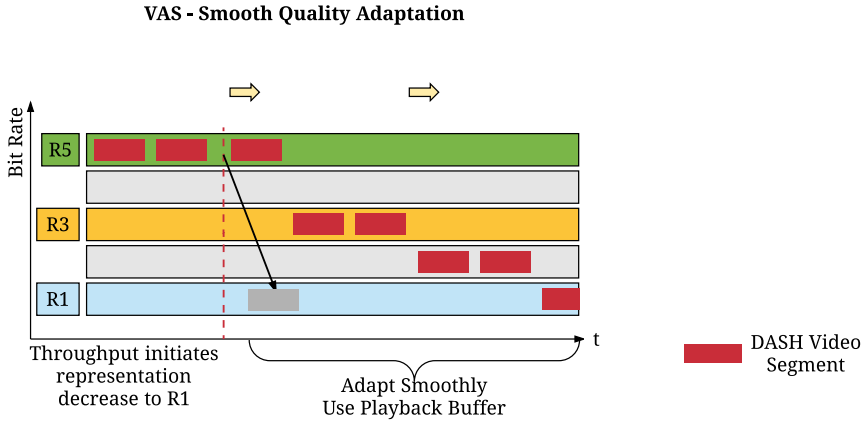



Figure 60: Example of using SQA as proposed by the VAS.

In a second step, an adaptation model is chosen by VAS, which represents the perceived qualities between the current and target representations. All intermediate representations are skipped until the representation with the index $i$ is found which lies between the source representation $R_S$ and target representation $R_T$. Thus, for an adaptation with quality decrease, the following condition must hold: $R_S > R_i > R_T$.

The representation $i$ is chosen so that the quality difference is nearly unnoticeable: $Q_{R_i} - Q_{R_S} < Q_{ND}$, where $Q_{ND}$ is the MOS value that defines a barrier when an adaptation between two representations is perceivable for the given content characteristics. This step is repeated for as many intermediate adaptations as necessary. The generalized function can be represented $Q_{R_i} - Q_{R_{RLV}} < Q_{ND}$, where $R_{RLV}$ represents the representation which was visited in the last planned adaptation. The $Q_{ND}$ values for different combinations are backed by the dataset provided by Moorthy et al. [**Moorthy2012**]. In the experiments, $Q_{ND}$ usually ranges between 0.05 and 0.19.

Ideally, this approach results in a list of unnoticeable adaptations between a source representation and a target representation, called the adaptation plan $P_{S,T}$. As those adaptations can be triggered by a decrease in network throughput, the danger of stalling is imminent if too many adaptations are conducted over a time period. As stalling events usually severely degrade the perceived quality, it is the aim to avoid their occurrence during a streaming session [**VanKester2011**]. A time constraint is defined based on the available buffer fill state ($FR_B$), the measured throughput ($TP_t$) and the bit rate of the current MPEG DASH representation ($BR_{R_i}$):

$$t_{P_{S,T}} = (FR_B - B_c * t_{B_{max}}) \times (1 + \frac{TP_t}{BR_{R_i}}) \quad \text{in} \quad [s] \tag{40}$$

Here, $B_c$ represents the fraction of the complete buffer length ($t_{B_{max}}$) that is preserved to ensure continuous playback.

For each adaptation at index $j$ in $P_{S,T}$ the time of the adaptation $t_j$ is determined by:

$$t_j = t + j * \frac{t_{P_{S,T}}}{N_{P_{S,T}}} \tag{41}$$

$N_{P_{S,T}}$ is the number of necessary adaptations.

Adaptations are distinguished between those that aim at decrease the quality level and those that increase it. The aforementioned approach is valid for decreasing adaptations. Following the findings of Hossfeld et al. and Moorthy et al., VAS aims to stay at

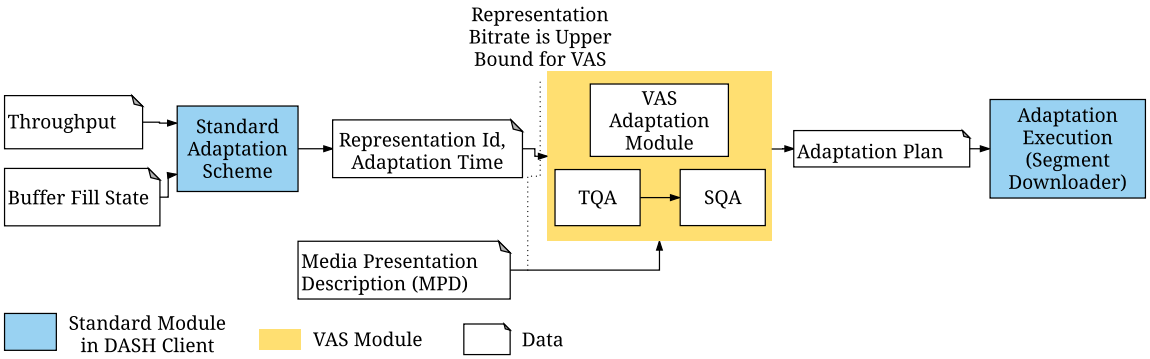



Figure 61: Integration of VAS adaptation schemes.

high-quality representations as long as possible but also smoothly adapt to lower representations [**Hossfeld2014**, **Moorthy2012**]. Furthermore, Lewcio et al. show that a quality-increasing adaptation improves the streaming experience [**Lewcio2011**]. Thus, VAS performs a direct jump from the source to the target representation for a quality-increasing adaptation.

### 7.4.3 *Integration into Existing Adaptation Schemes*

The concept of VAS is to support existing adaptation schemes that solely consider the current network conditions or the playback buffer level, in adapting in a content- and quality-aware manner. As mentioned, VAS does not introduce a new throughput-related adaptation scheme, but it can be plugged into existing adaptation modules of MPEG DASH streaming clients (as depicted in Figure 61).

The decision of the adaptation scheme introduced by the MPEG DASH client is used as an upper bound for the VAS decision, i.e., the maximum bit rate to be selected by the VAS adaptation scheme. The decision of which representation to choose is based on comparing throughput conditions of the network, the current playback buffer fill state, or a combination of both metrics. As a result, one representation, and an optional timestamp when to switch, are the output of VAS to the client.

The client's VAS component leverages information of the bit rate of the recommended adaptation and makes it an upper bound. The upper bound is known as $r_{t,max}$, where $r$ represents the representation index, $max$ depicts the highest representation index to be selected, and $t$ depicts the current timestamp. The adaptation plan generated by VAS does not include any representation with a higher bit rate; however, VAS first applies the TQA to select a representation with a lower bit rate which offers the desired quality level of the streaming session or an equal (or higher quality) than $r_{t,max}$. As the quality may change over upcoming video shots, VAS gives an adaptation plan to the adaptation execution component, which may consist of a list of representations to switch to and the adaptation timestamps.

## 7.5 EVALUATION OF THE VAS

The evaluation of VAS is structured into two parts. First, the possible data traffic reductions and then the quality gains when applying VAS's adaptation schemes are evaluated in a

prototypical evaluation setup. Then, subjective user studies are used to assess the impact of the proposed adaptation schemes on the streaming experience.

### 7.5.1 *Objective Analysis of VAS's Adaptation Schemes*

The data required for streaming video is determined in this section for VAS's adaptation schemes in comparison to state-of-the-art algorithms and the optimal adaptation scheme proposed in Section 7.4.1.

#### 7.5.1.1 *Setup of the Evaluation*

This section describes the experimental setup using a prototypical implementation of the concepts discussed in previous sections. The first paragraph of this section shows the physical setup, with mobile devices receiving video streams as well as network traces used for mimicking different real streaming sessions.

To mimic a real deployment of VAS as a support service, different state-of-the-art adaptation schemes are implemented for the MPEG DASH reference client, DASH.js[3]. The used adaptation schemes are discussed in the second paragraph.

The third paragraph describes the used MPEG DASH videos with different content characteristics specific for this evaluation.

*Setup of the Network Experiments*

The evaluation setup consists of the prototypical implementation of VAS, a video hosting server, and three mobile devices (Google Nexus 5) that receive the video streams. Wireless connections are available between all devices using an IEEE 802.11g access point. Mobile Android devices allow us to use the full network stack of a mobile device. The setup is depicted in Figure 62.

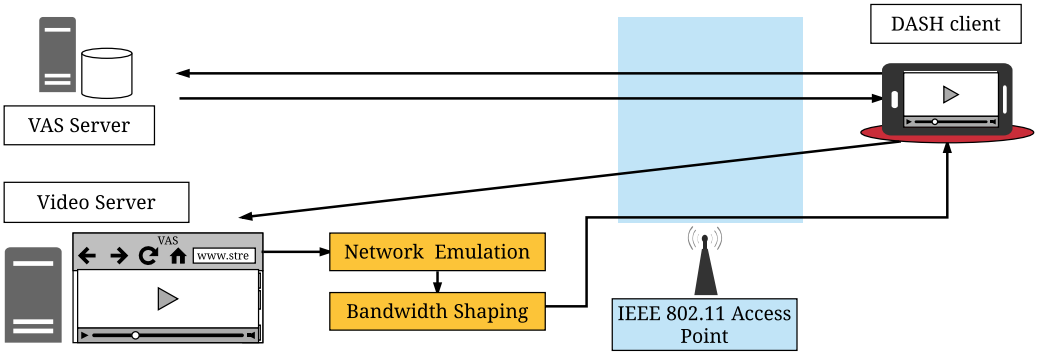



Figure 62: Evaluation setup for the VAS which shapes the data traffic for video segment delivery.

A single Ubuntu 14.04 server is used to host a web server that provides VAS and various videos. Even though the web server and VAS run on the same physical machine, they do not exchange any information locally. The quality models are generated online using the RT-VQM algorithm, which is run on three different Amazon Web Service (AWS) machines

3 https://github.com/Dash-Industry-Forum/dash.js; Visited on: 06/30/2016

of type "g2.2xlarge"[4] [**Wichtlhuber2016**]. Using the IEEE 802.11g access point, any VAS-enabled client can communicate without limitations with VAS.

Requests to the web server are not restricted, but MPEG DASH segments underlie a traffic shaping component for both the respective throughput and the latency. To mimic real streaming sessions and allow reproducible results, openly available network traces gathered by Eittenberger et al. and Riiser et al. are input for a traffic-shaping component on the server [**Eittenberger2013**, **Riiser2013**]. The throughput and latency on the path from the video server is limited on a per-client basis. Eittenberger et al.'s traces are based on monitoring YouTube streaming sessions in the UMTS network in the area of Bamberg, Germany. The network traces are sampled in an interval of 10 seconds in the network of the mobile operator T-Mobile using two different mobile Android devices (Huawei S7-301u MediaPad, Samsung Galaxy Tab GT-P1000). Two traces include video streaming sessions of a moving device with a speed of up to 50 km/h (TM1 and TM2) on a track of 3.6 km in the city center. The average throughput of the trace is 0.258 $\frac{\text{MBit}}{\text{s}}$ for TM1 and 0.822 $\frac{\text{MBit}}{\text{s}}$ for TM2. Two other traces are created with static device positions that show average throughputs of 0.194 $\frac{\text{MBit}}{\text{s}}$ for TS1 and 0.569 $\frac{\text{MBit}}{\text{s}}$ for TS2. One additional, synthetic trace with a constant throughput of 90 $\frac{\text{MBit}}{\text{s}}$ (TO1) was added, which depicts mobile streaming under ideal conditions.

The network traces of Riiser et al. [**Riiser2013**] include mobile streaming sessions in 3G networks in Northern Europe. The sessions are captured using the proprietary Opera Netview Media Client on Laptops using a Huawei Model E1752 HSPA USB stick, while streaming video from a dedicated video server. The trace is sampled approximately every second. The video streaming sessions used a HAS protocol for streaming video. Different vehicles are used including, bus, metro, tram, and car. The traces' average throughputs are depicted in Table 21.

For each evaluation run, a starting point of the trace is randomly selected while keeping the trace's characteristics. Ten evaluation runs are conducted for each video sequence and trace while its main characteristics - the average throughput and variance - deviate only slightly. At any time, a minimal throughput of 1 $\frac{\text{KBit}}{\text{s}}$ is guaranteed. The traces are tailored so that the throughput shaping is refreshed every 200 milliseconds. If the streaming session lasts longer than the available trace duration, the trace is repeated.

*Adaptation Schemes*

VAS is designed to support any existing MPEG DASH adaptation scheme, which fulfills the requirements discussed in Section 7.4.3. For making adaptation decisions, those schemes rely either on playback buffer information or a measurement of the current application-layer throughput. Schemes using the playback buffer analyze the playback buffer fill state, i.e., the ratio between the available video segments in the playback buffer in seconds and the size of the playback buffer in seconds.

Different state-of-the-art schemes of both categories are implemented. These are selected either from an experimental study of Thang et al. or from widely used MPEG DASH streaming clients [**Thang2014**]. The *Threshold-based Buffer Adaptation (TBB)* scheme assumes that a playback buffer is split into zones, which are defined by thresholds that initiate different adaptation behaviors. The $T_{B_c}$ represents the critical zone of the buffer, $T_{B_l}$ defines the zone in which the buffer fill state is perceived as low; $T_{B_n}$ is the desired buffer fill state and $T_{B_{max}}$ shows the buffer capacity. In this evaluation, the approach of Miller et al. is

4 Intel Xeon E5-2670 CPU with 16GB of memory and dedicated NVIDIA GRID GPUs with 1536 Compute Unified Device Architecture (CUDA) cores.

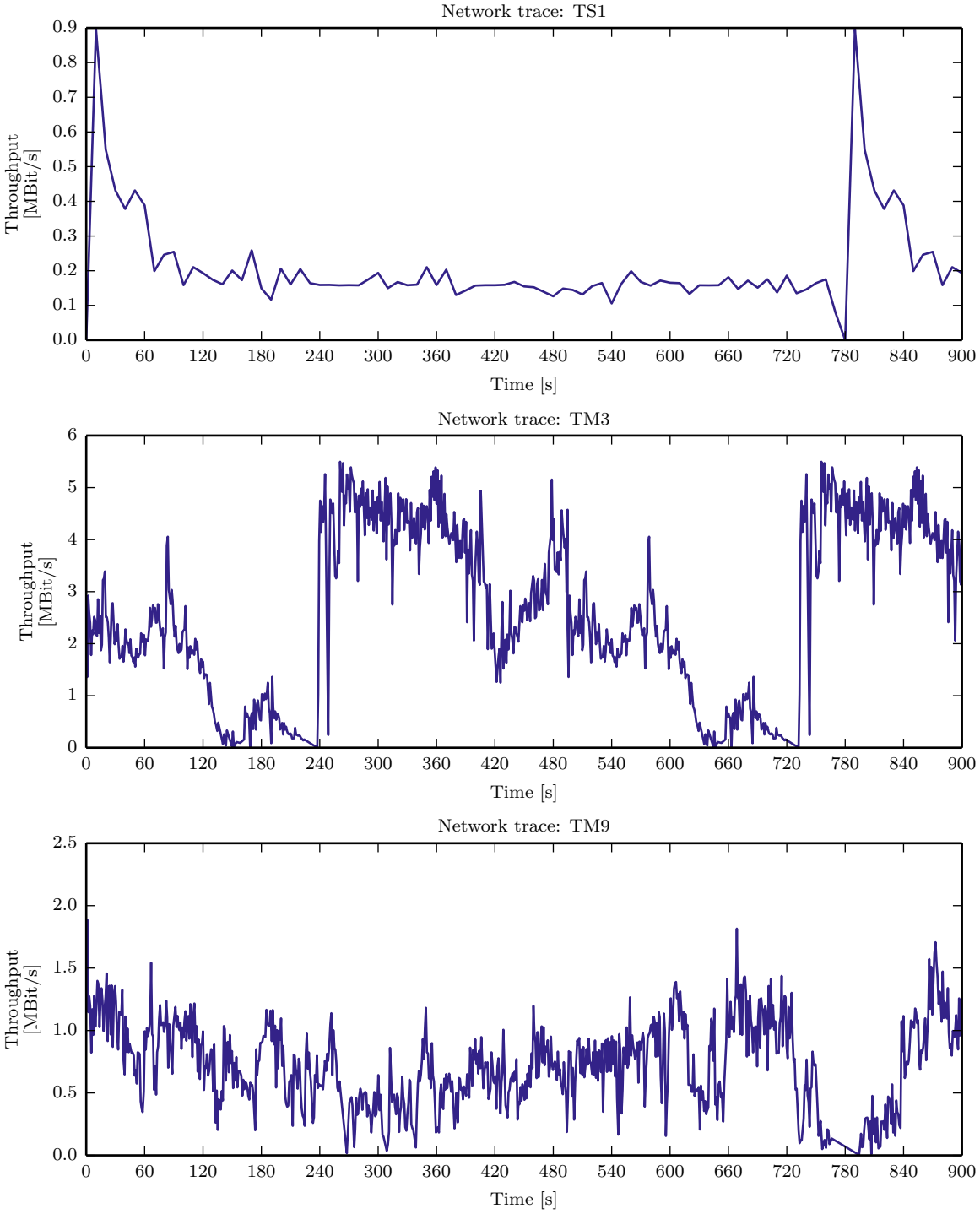



Figure 63: Network traces TS1 from Eittenberger et al.'s dataset [**Eittenberger2013**] and TM3, TM9 of Riiser et al. [**Riiser2013**] are shown for the first 900 seconds of a streaming session.

used in which a buffer fill state between $T_{B_c}$ and $T_{B_l}$ adapts to the next lower representation index [**Miller2012**]. A buffer fill state below $T_{B_c}$ initiates an adaptation to the lowest representation. The adaptation scheme maintains the current representation as long as the buffer fill state is between $T_{B_l}$ and $T_{B_n}$, while the representation index is increased when the buffer fill state is above $T_{B_n}$.

A *Throughput-based Adaptation (TB)* measures the current throughput of the network on the application layer and decides when and how to adapt based on this measure-

Table 21: Overview of the network traces used for evaluating VAS.

| ID | Application | Category | Duration | Avg. throughput |
|---|---|---|---|---|
| **Eittenberger et al. [Eittenberger2013]** | | | | |
| TM1 | YouTube | Mobile (Car) | 12:13 min | 0.258 Mbit/s |
| TM2 | YouTube | Mobile (Car) | 6:39 min | 0.822 Mbit/s |
| TS1 | YouTube | No Mobility | 12:42 min | 0.194 Mbit/s |
| TS2 | YouTube | No Mobility | 10:26 min | 0.569 Mbit/s |
| **Riiser et al. [Riiser2013]** | | | | |
| TM3 | Opera Netview | Mobile (Bus) | 8:14 min | 2.765 Mbit/s |
| TM4 | Opera Netview | Mobile (Bus) | 22:44 min | 2.617 Mbit/s |
| TM5 | Opera Netview | Mobile (Bus) | 9:47 min | 1.992 Mbit/s |
| TM6 | Opera Netview | Mobile (Bus) | 7:24 min | 2.447 Mbit/s |
| TM7 | Opera Netview | Mobile (Metro) | 3:15 min | 1.451 Mbit/s |
| TM8 | Opera Netview | Mobile (Metro) | 17:10 min | 0.584 Mbit/s |
| TM9 | Opera Netview | Mobile (Tram) | 23:48 min | 0.777 Mbit/s |
| TM10 | Opera Netview | Mobile (Tram) | 23:27 min | 0.921 Mbit/s |
| TM11 | Opera Netview | Mobile (Tram) | 25:11 min | 0.679 Mbit/s |
| TM12 | Opera Netview | Mobile (Tram) | 21:13 min | 0.791 Mbit/s |
| TM13 | Opera Netview | Mobile (Tram) | 25:09 min | 0.838 Mbit/s |
| TM14 | Opera Netview | Mobile (Ferry) | 19:22 min | 1.568 Mbit/s |
| TM15 | Opera Netview | Mobile (Ferry) | 9:15 min | 1.829 Mbit/s |
| TM16 | Opera Netview | Mobile (Car) | 123:20 min | 0.727 Mbit/s |
| TM17 | Opera Netview | Mobile (Car) | 203:44 min | 0.726 Mbit/s |
| TM18 | Opera Netview | Mobile (Car) | 7:16 min | 1.761 Mbit/s |
| TM19 | Opera Netview | Mobile (Train) | 40:42 min | 1.393 Mbit/s |
| TM20 | Opera Netview | Mobile (Train) | 36:42 min | 1.123 Mbit/s |
| **Artificial** | | | | |
| TO1 | - | No Mobility | 60:00 min | 90 Mbit/s |

ment [**Thang2012**]. The MPEG DASH representation to download is determined by its bit rate, which should be equal or below the current throughput of the network. The TB may lead to disturbing oscillation effects if the measured throughput suffers from adaptation oscillations. An approach to mitigate peaks in the measured throughput is to use the *Smoothed Throughput-based Adaptation (ST)* as: $TP_{smooth}(t) = ((1-\rho) * TP(t-1) + \rho * TP(t)) * (1-\beta)$, where $\rho$ represents the weight between $t-1$ and $t$ and $\beta$ is a safety margin. $t$ and $t-1$ are measured as the average throughput after the download of a complete MPEG DASH segment.

Combinations of throughput-based and buffer-based adaptation schemes are proposed, e.g., with Miller's *Buffer-based Throughput Adaptation (BTR)* [**Miller2012**]. BTR triggers adaptations according to the thresholds of the TBB, but determines the representation to switch to depending on the current throughput.

Akhshabi et al. show another hybrid method relying on the ST and the buffer levels $T_{B_c}$ and $T_{B_l}$, which is called *Threshold- and Smoothed Throughput-based Adaptation (TBST)* [**Akhshabi2011**]. For a buffer fill state below $T_{B_c}$ the lowest representation is chosen. In contrast, an adaptation to the next lower bit rate is performed if the measured throughput is below the current representation bit rate and the buffer fill state is between $T_{B_c}$ and $T_{B_l}$. Similarly, when above $T_{B_l}$ the representation index is increased by one, if the smoothed throughput is stable or increasing and the next representation's bit rate is below the smoothed throughput. This adaptation scheme smoothly adapts when the available throughput changes. The method is called threshold-based smoothed throughput measurement adaptation (TBSTA).

Müller et al. propose a hybrid model (Aggressive Threshold-based Adaptation (ATB)) [**Muller2012**]. ATB performs an aggressive adaptation where a buffer fill state below $T_{B_c}$ triggers an adaptation to a representation with a bit rate which is below 0.3 times the current throughput. A buffer fill state between $T_{B_c}$ and $T_{B_l}$ defines the desired bit rate to be below half of the measured throughput, and a fill state between $T_{B_l}$, and $T_{B_n}$ adapts to a bit rate equal or be-

low the current throughput. Above $T_{B_n}$, the current throughput is multiplied by $\gamma$, where $\gamma > 1$, to determine the next representation.

The last scheme discussed is the QoE-aware DASH Adaptation (QD) [**Mok2012**], which aims for quality-aware streaming but neglects content characteristics during adaptation decisions. To improve the perceived quality, QD [**Mok2012**] decreases the representation index if network conditions degrade. This behavior follows the same idea as the proposed SQA scheme but is rather static, as the intermediate representation is chosen to be one index above the target representation. If the network conditions allow a higher representation than the current one, a direct jump to the possible representation index is performed.

All the above adaptation schemes are evaluated with and without support of the VAS adaptation support. The optimal adaptation scheme is implemented using the optimization software Gurobi 6.5.2[5].

*Video Dataset*

The videos used for evaluation are from the publicly available MPEG DASH datasets provided by Lederer et al. consisting of the sequences: The Swiss Account (VTSA), Big Bucks Bunny (VBBB), Valkaama (VV), Of Forest and Men (VOFM), Redbull Playstreets (VRBPS), Tears of Steel (VTOS) and the Elephant's Dream (VED) [**Lederer2013**]. The characteristics of frame rate, bit rate in $\frac{\mathrm{KBit}}{\mathrm{s}}$, and resolutions for different MPEG DASH representations are shown in Table 22.

Table 22: Video encoding profiles of different videos from the MPEG DASH dataset [**Lederer2013**] with static frame rates for each video and variable resolutions (S) and bit rates (B) in $\frac{\mathrm{KBit}}{\mathrm{s}}$. R shows the MPEG DASH representation index.

| | **VBBB** 24FPS | | **VED** 24FPS | | **VOFM** 24FPS | | **VTOS** 24FPS | | **VTSA** 30FPS | | **VV** 30FPS | | **VRBPS** 30FPS | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| R | **S** | **B** | **S** | **B** | **S** | **B** | **S** | **B** | **S** | **B** | **S** | **B** | **S** | **B** |
| 0 | 240p | 46 | 240p | 46 | 240p | 47 | 270p | 255 | 240p | 91 | 240p | 46 | 240p | 101 |
| 1 | 240p | 89 | 240p | 91 | 240p | 91 | 360p | 508 | 240p | 131 | 240p | 88 | 240p | 151 |
| 2 | 240p | 131 | 240p | 131 | 240p | 135 | 360p | 811 | 360p | 174 | 240p | 123 | 360p | 201 |
| 3 | 360p | 178 | 360p | 180 | 360p | 186 | 544p | 1113 | 360p | 216 | 360p | 175 | 360p | 251 |
| 4 | 360p | 222 | 360p | 222 | 360p | 232 | 720p | 1516 | 360p | 257 | 360p | 214 | 360p | 301 |
| 5 | 360p | 263 | 360p | 261 | 360p | 277 | 720p | 2427 | 360p | 337 | 360p | 250 | 480p | 501 |
| 6 | 360p | 334 | 360p | 328 | 360p | 366 | 1080p | 3020 | 480p | 431 | 360p | 310 | 480p | 600 |
| 7 | 360p | 396 | 360p | 382 | 480p | 462 | 1080p | 4028 | 480p | 602 | 480p | 439 | 480p | 700 |
| 8 | 480p | 522 | 480p | 523 | 480p | 553 | 1080p | 6045 | 480p | 764 | 480p | 516 | 480p | 896 |
| 9 | 480p | 595 | 480p | 594 | 480p | 644 | 1080p | 10068 | 480p | 967 | 480p | 587 | 480p | 1180 |
| 10 | 720p | 791 | 720p | 796 | 576p | 825 | | | 720p | 1318 | 720p | 812 | 720p | 1499 |
| 11 | 720p | 1033 | 720p | 1033 | 576p | 1006 | | | 720p | 1727 | 720p | 989 | 720p | 1994 |
| 12 | 720p | 1245 | 720p | 1231 | 576p | 1268 | | | 720p | 2056 | 720p | 1237 | 720p | 2475 |
| 13 | 720p | 1547 | 720p | 1495 | 576p | 1519 | | | 1080p | 2714 | | | 1080p | 2996 |
| 14 | 1080p | 2134 | 1080p | 2118 | 576p | 1765 | | | 1080p | 3500 | | | 1080p | 3993 |
| 15 | 1080p | 2484 | 1080p | 2445 | 576p | 2159 | | | 1080p | 3995 | | | 1080p | 4982 |
| 16 | 1080p | 3079 | 1080p | 2980 | 576p | 2529 | | | | | | | 1080p | 5943 |
| 17 | 1080p | 3527 | 1080p | 3431 | 576p | 3206 | | | | | | | | |
| 18 | 1080p | 3840 | 1080p | 3791 | | | | | | | | | | |

These sequences are encoded using H.264/AVC, with a stable frame rate but varying resolutions and quantization parameters [**Wiegand2003**]. The described dataset represents the common number of representations and encodings available for MPEG DASH streaming sessions, e.g., on YouTube or Netflix. Content Delivery Networks (CDNs) distribute even more representations than used in the MPEG DASH dataset. These representations

5 http://www.gurobi.com/index; Visited on: 06/30/2016

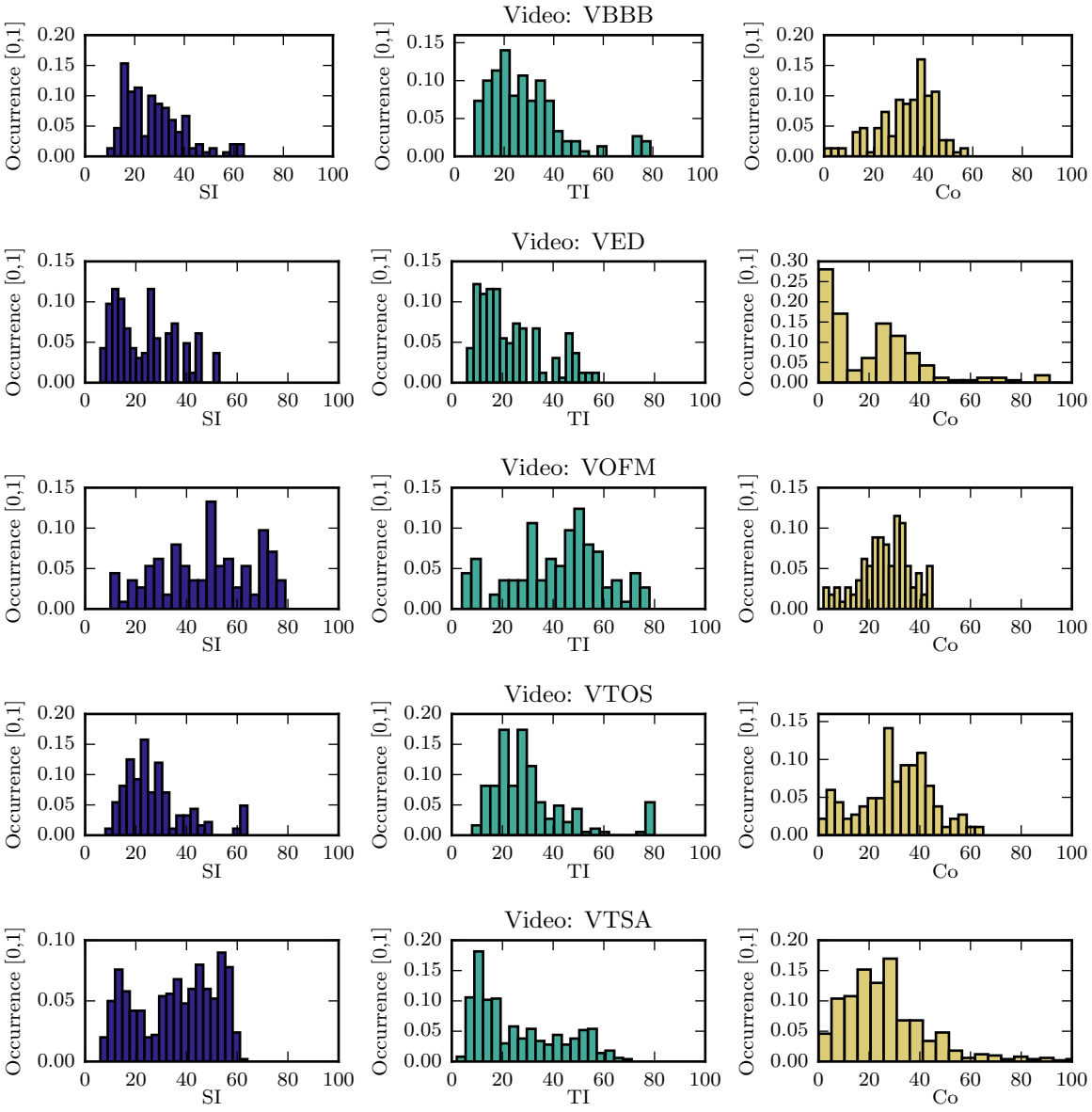



Figure 64: Classification of test videos regarding SI, TI, and Co.

differ in terms of their bit rates, resolutions, and frame rates and are available for different device categories [**Krishnappa2015**]. Therefore, we transcode a video sequence into 60 representations with varying frame rates, resolutions and quantization levels. The result is the MPEG DASH video VHEVC, which is encoded in all combinations of spatial dimension (resolutions: 180p, 360p, 540p, 720p, 1080p), temporal dimension (frame rate: 15, 30, 45, 60 FPS ) and quality dimension (quantization: 21, 27, 33). The bit rate of the lowest representation is 116.504 $\frac{\text{KBit}}{\text{s}}$, the highest representation is 26.71 $\frac{\text{MBit}}{\text{s}}$.

The video set used for evaluation offers a broad variety of structure, motion, and color information in order to show that VAS achieves significant data traffic savings for different videos. The videos differ in terms of the content features SI, TI, and Co. Figure 64 shows histograms on a subset of five videos (VBBB, VED, VOFM, VTOS, and VTSA). The features are calculated per video shot and the occurrence of the feature values are normalized between 0 and 1. Figure 64 illustrates that the video shots cover a broad range of the feature spectrum.

#### 7.5.1.2 *Data Traffic Reduction*

This section describes the achieved data traffic reductions using VAS for different adaptation schemes. The metric used for assessing data traffic reductions is the relative data traffic generated ($Q = \frac{DT_{VAS,AS}}{DT_{AS}}$), where DT represents the amounts of bits transferred. $VAS, AS$ indicates that a VAS scheme is used, whereas $AS$ indicates a non-VAS scheme. The relative data traffic reductions are expressed as $R = 1 - Q$.

*Different Adaptation Schemes*

Using VAS allows streaming clients to specify a target quality for adapting video. In contrast to bit rate-based adaptation, this does not mean that the highest bit rate representation is always streamed. This offers the potential to save data traffic.

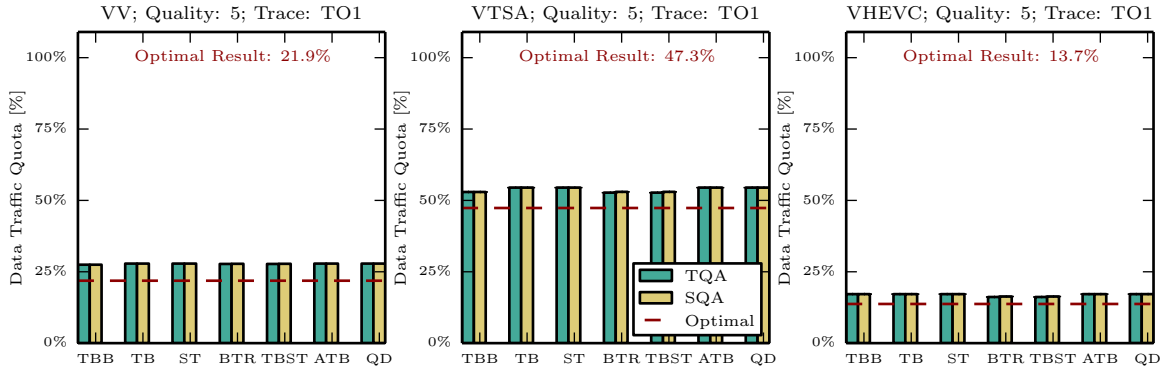
A first scenario describes how much data traffic can be saved for a common mobile streaming client in an overcapacity scenario (network trace: TO1) for all videos. For TO1, all evaluated schemes will eventually adapt to the highest bit rate representation, while VAS solely adapts towards the lowest bit rate representation offering the desired quality. For this scenario, the target quality is set to an MOS of 5, representing the highest available quality of a video stream. The remaining parameters are kept static for this evaluation: HTTP Partial Get: Off, Playback Buffer Length: 20 seconds, Segment Duration: 2 seconds, and Session Time: 60 minutes[6].

The results depicted in Figure 65 show achieved data traffic quota (Q) for the three video sequences VTSA, VRBPS, and VHEVC - and all adaptation schemes. Also, the optimal adaptation is depicted as a red, dotted line.

Obviously, in the overcapacity scenario the data traffic quotas for all adaptation schemes are similar, but the data saving potential is very diverse for different videos. As in an overcapacity scenario, none of the adaptation schemes invokes a quality decreasing adaptation. Thus, TQA and SQA achieve a similar data traffic quota (Q). For the evaluated MPEG DASH video sequences (see Table 22), the data traffic quota is in the range of 17.17% (HEVC) to 72.64% (VRBPS), meaning that even for VRBPS at the highest target quality level, the data saving potential lies at around 27.36%. For videos similar to HEVC that have a multitude of representations, different video dimensions allow adaptation across frame rate, resolution, and quantization level. Those adaptation schemes achieve data savings of up to 82.83%. As external conditions are similar for all videos, data traffic ratio differences are related to the encoded content, especially the volatility of the content characteristics. Especially, highly varying video sequences such as HEVC, VTOS, or VV achieve large data savings, whereas VOFM or VRBPS have a low number of video shots that are usually rather long. VRBPS achieves the smallest reduction of around 27.36% due to the variability of motion, structure, and color characteristics being comparably low between different video shots (see Figure 64). This effect is related to higher encoding efficiency if video characteristics stay similar over longer times. VAS heuristics, such as TQA, benefit from videos with more representations, and thus, more fine-granular adaptation options. Another reason for high data traffic savings is the availability for high bit rate and resolution representations. For example, the low maximum resolution and bit rate of VOFM (576p, 4 $\frac{MBit}{s}$) limits VAS's potential for data traffic reduction.

Also, the optimal adaptation is depicted for the videos VHEVC, VRBPS, and VTSA in Figure 65, and for all videos in Table 23. It is shown that the achieved data traffic quota of

6 The segment length has been studied with values of 2 seconds, 4 seconds, 6 seconds and 10 seconds and the playback buffer length of 4 seconds, 12 seconds and 20 seconds. Small but significantly improved results could be achieved using a small segment length, i.e., 2 seconds, and a large playback buffer size (20 seconds).




(a) (b) (c)

Figure 65: Overview of the data traffic quota of VAS's TQA and SQA in relation to the pure adaptation scheme without VAS for the network trace TO1. The plots represent the videos (a) VV, (b) VTSA, and (c) VHEVC.

the adaptation scheme ATB is close to the optimal adaptation. Other adaptation schemes show similar results for the overcapacity trace (TO1). On average, the difference between the proposed heuristics and the optimal adaptation in the overcapacity case ranges from 3.47% to 7.14%. VAS's heuristics are close to the optimal results achieved with global knowledge of the network trace and video characteristics. Differences can be explained as the adaptation schemes supported by VAS slowly adapt up to the highest quality representation in the beginning, whereas the global knowledge of the optimal adaptation allows an immediate switch to the highest quality. VAS's gap to the optimal adaptation is different for each video, as video representations differ in terms of bit rates, and VAS is not aware of the video characteristics of future MPEG DASH segments. Again, VAS heuristics such as TQA benefit from videos with more representations and adaptation options in all three video dimensions. In the case of the HEVC sequence, the 60 encoded representations show a lower relative gap between the heuristics and the optimal adaptation.

Table 23: Data traffic quotas when VAS is applied to an adaptation scheme for different video sequences in comparison to the optimal adaptation model (Trace: TO1).

| | VBBB | VED | VOFM | VTSA | VV | VRBPS | VTOS | VHEVC |
|---|---|---|---|---|---|---|---|---|
| ATB supported by VAS | 46.08% | 45.01% | 55.04% | 52.95% | 27.48% | 72.63% | 31.83% | 17.17% |
| Optimal Adaptation | 40.5% | 39.4% | 47.9% | 47.3% | 21.9% | 65.6% | 26.2% | 13.7% |
| Difference | 5.58% | 5.61% | 7.14% | 5.65% | 5.58% | 7.03% | 5.63% | 3.47% |

Significant data traffic savings can be achieved in an overcapacity scenario, but mobile streaming clients often suffer from either a lack of throughput or highly fluctuating throughput scenarios. The TS1 and TS2 traces were recorded by mobile devices without mobility in UMTS networks, and TM1 to TM20 were recorded in different moving vehicles in UMTS networks. Challenging network conditions will lead to different behaviors of the adaptation schemes. Thus, different data saving potentials were observed.

Figure 66 depicts the results of applying VAS on the network trace TM9, which contains an average throughput of only 777 $\frac{\text{KBit}}{\text{s}}$. This is not a sufficient throughput for any of the video sequences to stream the highest bit rate representations at all times.

Note that Figure 66 shows the data traffic reductions of TQA and SQA. The differences are minimal as SQA is only invoked if an adaptation to a lower quality level is required. In

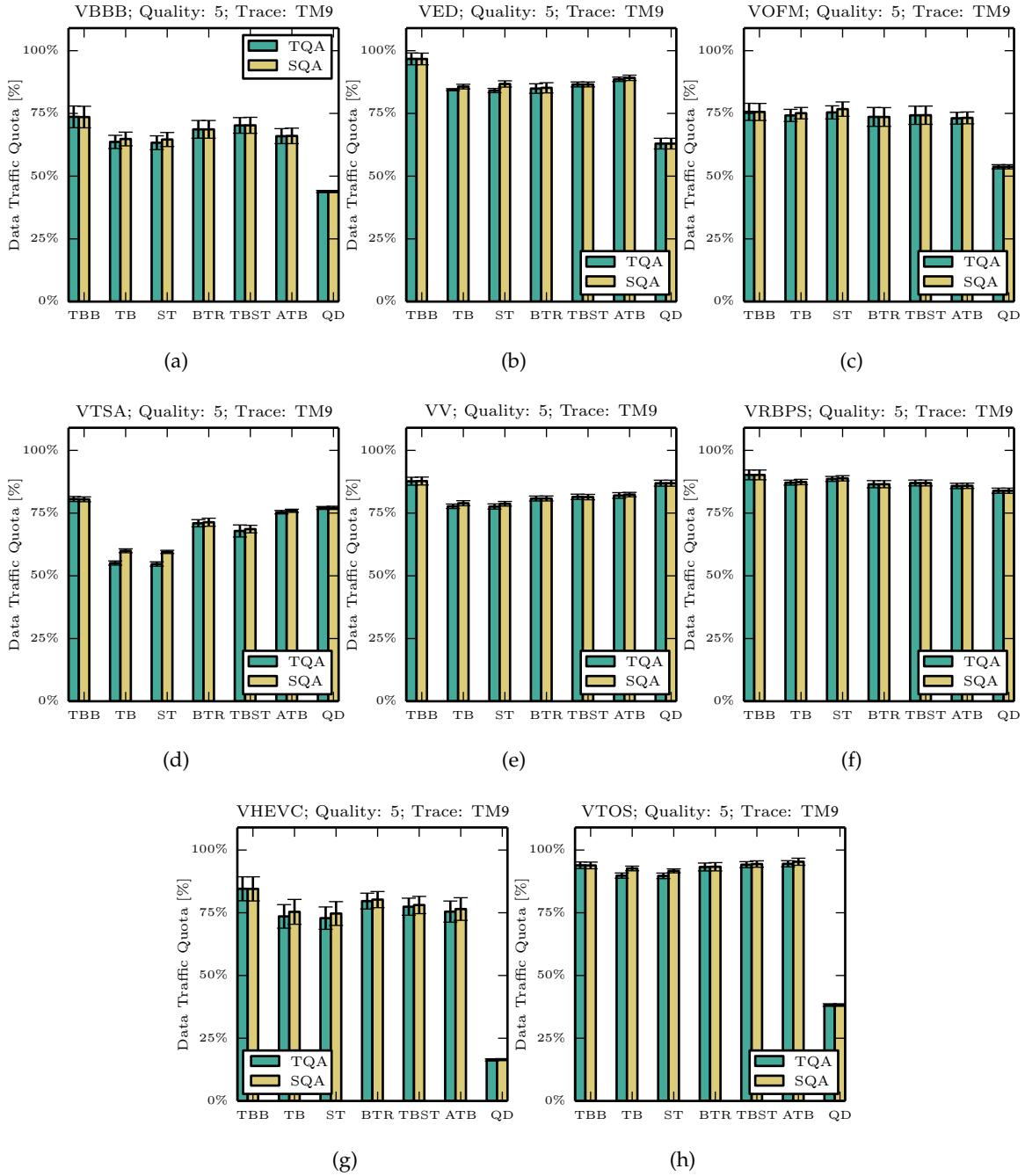



Figure 66: Data traffic quota of using VAS's TQA and SQA in relation to the pure adaptation scheme for a trace in a challenged network: TM9. The aim is to stream the highest available quality (MOS: 5). The different plots represent the videos VBBB, VED, VOFM, VTSA, VV, VRBPS, VHEVC, and VTOS.

comparison to the overall streaming session length, SQA is only active for a rather short time. Thus, only small differences of TQA and SQA are observed. A detailed discussion of the SQA scheme is given in Section 7.5.1.3.

VAS TQA scheme achieves data traffic reductions, even under these challenged conditions, of at least 3.32% for adaptation scheme TBB and the sequence VED, and up to 84.4% for the video sequence VHEVC and the adaptation scheme QD. For most of the videos (VBBB, VED, VOFM, VRBPS, VHEVC, and VTOS), QD benefits most from the application

of VAS. This can be explained, as in comparison to the other schemes QD risks stalling in a video for the sake of streaming a high quality representation. Thus, in challenged network conditions QD requests higher bit rate representations. Thus, the potential for bit rate savings is higher when using VAS in comparison to other adaptation schemes. The other adaptation schemes benefit differently from the VAS.

Table 24 gives an overview on achievable data traffic savings for the TM3 trace, which has an average throughput of 2.7 $\frac{\mathrm{MBit}}{\mathrm{s}}$ and is highly fluctuating over time due to a bus ride with many throughput reductions losses (see Figure 63 (b)). Also, the table shows the average data traffic quota achieved for all network traces. Again, QD benefits most from applying VAS in the trace TM3 with data traffic quotas between 16.2% and 80.4%. For TM3 ATB shows the lowest data saving potential with a data traffic quota of 94% for the VRBPS sequence, and thus a saving potential of only 6%. Table 24 also depicts the average data traffic quotas for different videos (last row) and all adaptation schemes (last column) for all evaluation runs. The highest saving potential exists for the VHEVC (44%) and the lowest for VRBPS (19.6%). For all traces, QD has a data traffic quota of 62.3% on average, and has the highest potential to save data traffic (37.7%), whereas TBB achieves data saving of 19.2%. On average, VAS is able to achieve data traffic savings of 27.7% independent of

Table 24: Data traffic quota of VAS-supported and VAS-unsupported adaptation schemes for trace TM3 and aggregated over all traces for all video sequences. The variances or confidence intervals are not shown as the variance is below $10^{-3}$ for TM3 in all cases.

| | VBBB | VED | VOFM | VTSA | VV | VRBPS | VTOS | VHEVC | All Videos |
|---|---|---|---|---|---|---|---|---|---|
| TM3 | | | | | | | | | All Traces |
| TBB | 64.6% | 62.8% | 73.4% | 75.3% | 45% | 88.4% | 88.8% | 82.3% | 81.8% |
| TB | 57.6% | 55.7% | 70.5% | 54.9% | 40.4% | 85.5% | 89.4% | 69.7% | 68.5% |
| ST | 57.8% | 55.5% | 71.4% | 54.3% | 40.4% | 87.2% | 89.6% | 68.8% | 69% |
| BTR | 63.4% | 60.6% | 73.1% | 69.7% | 43.4% | 85.8% | 90.3% | 77.6% | 74.3% |
| TBSTA | 60.7% | 58.6% | 74% | 65.1% | 43.1% | 83.4% | 89% | 75.8% | 73.6% |
| ATB | 58.7% | 57.9% | 70% | 69.9% | 42% | 83.9 | 94% | 71.5% | 76.6% |
| QD | 44.6% | 42.8% | 53.3% | 71.8% | 45.8% | 80.4% | 36.8% | 16.6% | 62.3% |
| All Traces | | | | | | | | | |
| Avg. | 71.7% | 66.3% | 78.5% | 70.5% | 57.3% | 90.5% | 78.7% | 56% | **72.3%** |

the video sequence, adaptation schemes or network trace. One should note that the traces used in this evaluation represent challenging conditions for mobile video streaming. Under these difficult conditions, VAS achieves considerable data savings across video sequences and adaptation schemes. These savings are thus higher when inspecting, e.g., the recently available 2160p videos or network traces from mobile networks with high throughput rates.

*How does VAS achieve the savings?*

To better understand achieved data traffic savings, session playback of streaming video with and without VAS support is shown in Figure 67 for trace TO1. For the overcapacity scenario, the VAS-unsupported adaptation schemes switch quickly to the highest bit rate available (gray line), whereas VAS induces additional adaptations to stream the highest available quality at the lowest possible bit rate. The red area indicates the achieved data gains over time. For the trace TM3, VAS adapts accordingly and achieves considerable data savings, too. Another advantage of VAS is shown in the buffer statistics of the figure. Whereas the classical adaptation scheme runs out of buffer frequently, VAS achieves a higher average fill rate of the playback buffer for challenging network conditions (see Figure 67 b).

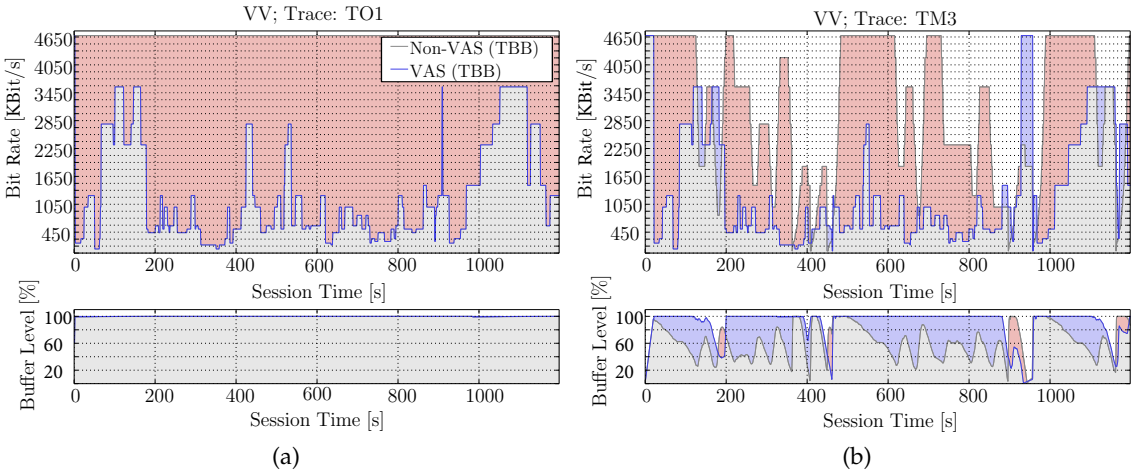



Figure 67: Comparison of the session traces of the video VBBB for the network traces: TO1 and TM3. In each plot the blue line depicts the bit rate streamed using VAS in comparison to the standard adaptation scheme. A red area indicates throughput savings of the VAS scheme. Plots at the bottom shows the buffer fill rate, where a blue area shows a higher buffer filling rate achieved by VAS.

This principle is illustrated in Figure 68 (a). The figure illustrates a comparison of the session times streamed on different MPEG DASH representations for the different adaptation schemes with or without VAS. A higher representation index represents a higher bit rate. It is obvious that in the overcapacity situation, the proportion of lower representations during streaming sessions increases for all videos. Even for the trace TM3, the time is reduced on higher index representations. At the same time, it already indicates (as in the case of TM3) that VAS uses saved data traffic to stream higher representations, when the standard adaptation scheme has to stream the lowest representation.

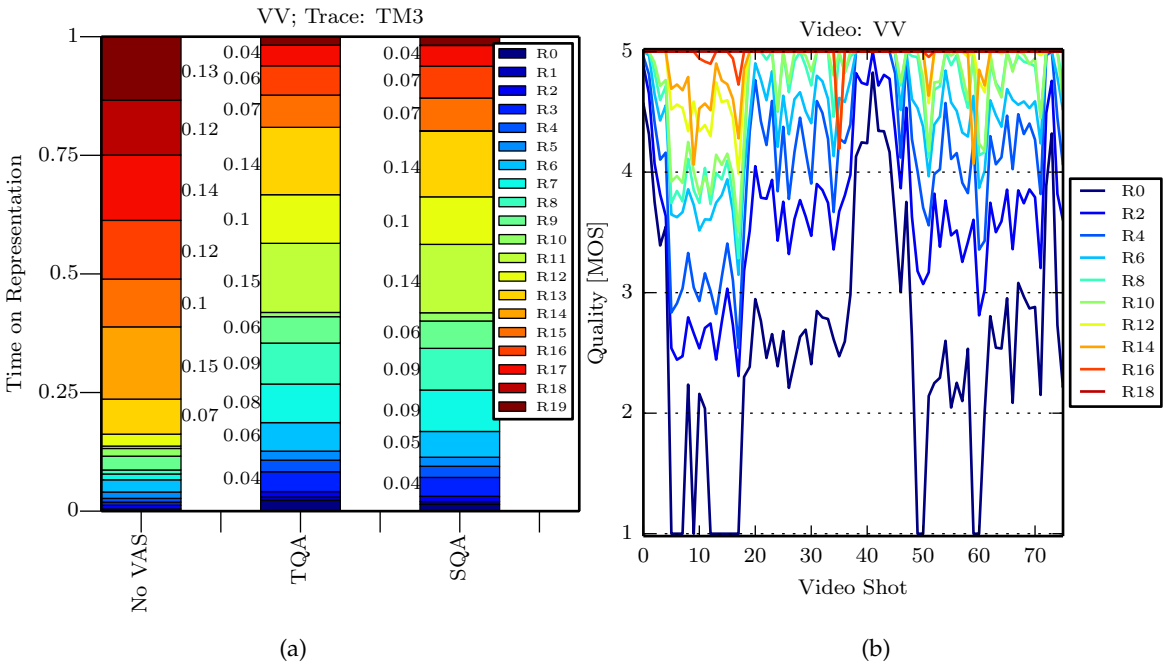
*Influence of SQA on Data Savings*

So far, the results discuss the TQA scheme that neglects the influence of SQA. It achieves a perceived quality gain by delaying quality-decreasing adaptations and thus staying longer at higher bit rate representations than TQA. In the case of TO1, SQA generates no additional traffic at all. For the remaining traces, e.g., as depicted for TM9 in Figure 66, an average data traffic quota increase of 0.7% can be observed.

Section 7.5.1.3 and Section 7.5.2 report on the advantages of SQA for the perceived quality of a streaming session.

*Segment-based Adaptations and Partial HTTP GET Requests*

An additional data traffic reduction can be achieved if the DASH client supports partial HTTP GET requests, thus requesting individual byte ranges from a video segment. This allows adaptation not only at segment boundaries, but also within a DASH segment, offering rich potential to adapt for either improved perceived quality or a data traffic reduction. Adaptations occur at GoP boundaries, but within a DASH segment. A shot boundary could lie within such a segment, resulting in different quality requirements for different parts of the DASH segments. A partial HTTP GET request can extract a byte range from a DASH segment that represents an independently decodable frame range, which is determined at encoding time by the GoP.




(a) (b)

Figure 68: Time on MPEG DASH representations when using VAS's TQA and SQA scheme for TM3 and video VV: (a) shows a reduced time on higher bit rate representations for all videos in an overcapacity situation (TO1), while (b) shows the achieved quality of different MPEG DASH representations over video shots.

Table 25 compares for streaming the videos VBBB, VED, VOFM, VTSA, and VHEVC using VAS TQA via HTTP 1.1/GET requests (adaptation at segment boundaries only) and partial HTTP 1.1/GET requests (adaptation within segments). When partial GET requests are used, results indicate a potential of 2.5% in additional data traffic savings. In none of the traces or video sequences did partial GET requests decrease the data saving potential. The impact on the perceived quality is discussed in Section 7.5.1.3. Partial HTTP GET requests can achieve lower data savings, when no potential for additional adaptations exist. The overhead for using VAS with partial HTTP GET is higher, as more requests are sent to the video server and the VAS server is consulted more often. Table 25 shows two examples. For video VV and the two adaptation schemes TB and ST the best result is achieved when streaming without partial HTTP GET requests.

Table 25: Data traffic quota of VAS-supported and unsupported adaptation schemes for trace TM3, aggregated over all traces for all video sequences. The table depicts the results for both the segment-based HTTP GET requests and partial HTTP GET requests separated by "/".

| Data Traffic Quota: HTTP GET / partial HTTP GET | | | | | | |
|---|---|---|---|---|---|---|
| | VBBB | VED | VOFM | VV | VHEVC | All Videos |
| Trace:TM3 and TQ: 5 | | | | | | All Traces |
| TBB | 64.6%/58.8% | 62.8%/57.4% | 73.4%/70.1% | 45%/41.9% | 82.3%/69.9% | 81.8%/76.3% |
| TB | 57.6%/57.8% | 55.7%/55.1% | 70.5%/70.8% | 40.4%/40.6% | 69.7%/68.1% | 70.6%/66.7% |
| ST | 57.8%/57.9% | 55.5%/55.2% | 71.4%/71.1% | 40.3%/40.7% | 68.8%/68.1% | 71.2%/67.4% |
| BTR | 63.4%/58.5% | 60.6%/56.7% | 73.1%/70.4% | 43.4%/41.9% | 77.6%/65.7% | 74.6%/73.5% |
| TBSTA | 60.7%/58.6% | 58.6%/56.7% | 74%/69.6% | 43.1%/41.8% | 75.8%/65.1% | 74%/73.7% |
| ATB | 58.7%/58.8% | 57.9%/56.8% | 70%/69.2% | 42%/41.7% | 71.5%/69.1% | 76.4%/73.5% |
| QD | 44.6%/44.3% | 42.8%/42.4% | 53.3%/53.2% | 45.8%/45.8% | 16.6%/15.7% | 62.3%/62.1% |
| All Traces and Quality:5 | | | | | | |
| Avg. | 71.7%/69.1% | 66.3%/64.1% | 78.5%/75.3% | 57.3%/55.7% | 56%/53.9% | **73%/70.5%** |

#### 7.5.1.3 *Achieved Quality during Streaming Sessions*

VAS aims at stabilizing the perceived quality during streaming while achieving significant data traffic reductions. For assessing the achieved quality the objective quality assessment metric RT-VQM is used [**Wichtlhuber2016**]. Note, that current video quality assessment metrics do not assess the impact of adaptations. Assessing the impact of adaptations is evaluated in Section 7.5.2.

This section describes the effect of VAS on the objectively estimated quality by comparing classical adaptation schemes with VAS TQA and SQA schemes, which describes both the influence of video shot-accurate adaptation using partial HTTP GET requests and the influence of SQA on stalling.

*Different Adaptation Schemes*

The results for determining the average quality per adaptation scheme are illustrated in Table 26. This shows an averaged, estimated quality value per adaptation scheme over all network traces and video sequences. Quality is measured in terms of MOS. The table describes the performance of each adaptation scheme with and without VAS support, but neglects additional quality impairments such as adaptation effects (see Section 7.5.2) and stalling (see Section 7.5.1.3).

It can be observed that significant differences in the qualities of the unsupported adaptation schemes exist. Adaptation schemes such as TBB, which represent buffer-based decision making, perform worst but in close range to the hybrid scheme BTR which represents an extension of TBB (difference in MOS: 0.02). Whereas TBB solely relies on buffer thresholds for deciding when and how to adapt, BTR answers the question of which representation to switch to using instant throughput estimates. Solely buffer-based adaptation schemes show the worst performance. A good performance in terms of improving the average quality of a streaming session is achieved by the adaptation schemes TB (MOS: 4.17), ST (MOS: 4.14), and ATB (MOS: 4.18). With TB and ST, two adaptation schemes solely relying on the throughput measurements are proposed, which outperform buffer-based adaptation schemes. Superior performance is achieved by hybrid models, which integrate both buffer-based and throughput-related adaptation decisions - as ATB already shows. Superior quality is achieved by TBST (MOS: 4.29) and QD (MOS: 4.69). Thus, QD and TBST especially indicate that considering both metrics, buffer fill state and throughput, can be beneficial. Additionally, QD adds a quality-aware adaptation scheme, which achieves an MOS increase of up to 0.4 in comparison to ATB. In Section 7.5.1.3 it is shown that QD in particular has significant problems to ensure a stall-free playback.

Table 26: Perceived quality using VAS-supported and unsupported adaptation schemes for the TM3 trace and aggregated over all traces for all video sequences. The variances or confidence intervals are not shown as the variance of the results is below 0.02.

| | HTTP GET | | | | Partial HTTP GET | | | |
|---|---|---|---|---|---|---|---|---|
| | Quality [MOS] | | | Traffic | Quality [MOS] | | | Traffic |
| | Standard | TQA | SQA | Quota | Standard | TQA | SQA | Quota |
| TBB | 4.08 | 4.36 | 4.35 | 81.8% | 4.03 | 4.41 | 4.42 | 76.3% |
| TB | 4.17 | 4.3 | 4.33 | 70.6% | 4.15 | 4.3 | 4.41 | 66.7% |
| ST | 4.14 | 4.32 | 4.36 | 71.2% | 4.26 | 4.34 | 4.41 | 67.4% |
| BTR | 4.1 | 4.29 | 4.33 | 74.6% | 4.18 | 4.37 | 4.37 | 73.5% |
| TBST | 4.29 | 4.31 | 4.32 | 74% | 4.31 | 4.39 | 4.39 | 73.7% |
| ATB | 4.18 | 4.48 | 4.5 | 76.4% | 4.27 | 4.54 | 4.63 | 73.5% |
| QD | 4.69 | 4.98 | 4.99 | 62.3% | 4.74 | 4.99 | 4.99 | 62.1% |
| | 4.23 | 4.43 | 4.45 | 73% | 4.28 | 4.47 | 4.52 | 70.5% |

VAS support improves the average MOS of all adaptation schemes, independent of whether the TQA or SQA scheme is used. Especially for challenged network traces, the saved data traffic in times with high throughput can be efficiently used to keep a high buffer fill ratio. For the adaptation schemes TBB, ATB, and QD, an MOS increase of approximately 0.3 is achieved, whereas TB, ST and BTR only benefit from an increase of 0.13 to 0.19. Both schemes benefit most, as they allow for a rapid increase in the representation index in situations when the measured throughput rises. The majority of network traces used for the evaluation show throughput increases and decreases due to mobility. In addition, both schemes slowly decrease representation indexes, thus allowing VAS to stream high quality representations over a longer period of time. By design, VAS never recommends a representation, which is higher than the one proposed by the pure adaptation scheme. The TBST scheme achieves the smallest improvement. TBST by design solely allows for small increases and decreases by one representation index. Especially for situations with a rapid increase in throughput, TBST only slowly improves quality. The buffer-based or purely throughput-reliant schemes achieve higher gains.

The introduction of partial HTTP GET requests offers the opportunity to not only adapt at MPEG DASH segment boundaries. On the one hand, this allows data traffic reduction by more accurately streaming the desired bit rate by switching within MPEG DASH segments - bound to an independently decodable GoP. On the other hand, this allows for improving quality when network conditions rapidly change. The adaptation control loop - measuring the buffer fill state, the throughput or both - is more frequently triggered and thus allows for adaptation decisions with a higher precision. Consequently, the results indicate an increase of 0.19 in comparison to the standard adaptation schemes at a significantly lower data traffic quota - i.e., higher data saving potential. In comparison to standard adaptation schemes without the support of partial HTTP GET requests, an improvement of around 0.29 is observed. Both results indicate a human-perceivable difference in the resulting video stream.

*Comparing TQA and SQA*

For most schemes (see Table 26), a rather small improvement can be observed using SQA in contrast to TQA; this is possible as no direct quality-decreasing switches are introduced but more time is spent on higher representations, which brings the risk of playback buffer depletion. The average effect for all adaptation schemes lies at approximately 0.02 in comparison to TQA. With the introduction of partial HTTP requests, SQA achieves a more granular adaptation option when decreasing representation indexes. As a consequence, MOS improvements rise from 0.02 to 0.05 on average, where especially TB, ST, and ATB benefit most with an average increase of 0.09 in comparison to 0.03 for classical HTTP GET.

Based on the aforementioned results: In comparison to TQA, a superior SQA performance can be observed for most adaptation schemes using HTTP GET requests and for all using partial HTTP GET requests.

*Influence of Stalling*

Previous discussions report on the advantage of VAS as measured in terms of perceived qualities averaged for the streaming session, neglecting stalling effects. Different adaptation schemes may more or less efficiently avoid playback buffer underruns. Thus, Table 27 depicts in the "Standard" column the average quality achieved when using an adaptation scheme and the impact of occurring stalls during streaming sessions as well as the resulting quality scores.

For estimating the impact of stalls, the model of Mok et al. [**Mok2011**] integrates the mean stalling time ($L_T$) and the frequency ($L_{FR}$) of stalls in a time window of about 90 seconds, and measures the impact in large subjective studies for HAS. The model was explained in Section 2.6.3.1.

Table 27 depicts the quality scores estimated by the objective quality assessment metric, the effect of stalling on the perceived quality and and the resulting quality scores. The table shows only the results of partial HTTP GET requests as it is superior for VAS-assisted adaptation. For standard adaptation schemes, an initial surprise is a significantly reduced quality achieved by QD. Even though it is designed to support quality-aware adaptation, its design does not reliably avoid stalling. It achieves streaming high quality representations at the cost of risking stalls. In contrast, a superior performance can be observed for ATB, ST, and TBST (MOS of 4.07, 4.07 and 4.12). In general, the stalling impact is rather low when averaged over all sessions. On average, the MOS decreases by 0.38 when averaged across all adaptation schemes.

Table 27: Influence of stalling proposed by Mok et al. on the quality scores for using VAS's TQA and SQA schemes in comparison to standard adaptation schemes. Results solely depict streaming sessions using partial HTTP GET results.

| | Quality [MOS] | | | Stalling Effect [MOS] by Mok [**Mok2011**] | | | Quality Score [MOS] | | |
|---|---|---|---|---|---|---|---|---|---|
| | Standard | TQA | SQA | Standard | TQA | SQA | Standard | TQA | SQA |
| TBB | 4.03 | 4.41 | 4.42 | 0.17 | 0.11 | 0.1 | 3.86 | 4.3 | 4.32 |
| TB | 4.15 | 4.3 | 4.41 | 0.19 | 0.08 | 0.08 | 3.96 | 4.22 | 4.33 |
| ST | 4.26 | 4.34 | 4.41 | 0.19 | 0.08 | 0.09 | 4.07 | 4.26 | 4.32 |
| BTR | 4.18 | 4.37 | 4.37 | 0.18 | 0.09 | 0.09 | 4.00 | 4.28 | 4.28 |
| TBST | 4.31 | 4.39 | 4.39 | 0.19 | 0.06 | 0.06 | 4.12 | 4.33 | 4.33 |
| ATB | 4.27 | 4.54 | 4.63 | 0.2 | 0.05 | 0.05 | 4.07 | 4.49 | 4.58 |
| QD | 4.74 | 4.99 | 4.99 | 1.51 | 1.4 | 1.42 | 3.23 | 3.59 | 3.57 |
| | 4.28 | 4.47 | 4.52 | 0.38 | 0.26 | 0.27 | 3.9 | 4.21 | 4.25 |

The resulting impact of stalling is reduced using VAS by an MOS of 0.12 when applying TQA and around 0.11 using SQA. These quality improvements are an additional side effect of VAS's bit rate savings, which can now ensure more stable playback of a video stream. In situations when the available throughput is scarce, any saved bit can be leveraged to ensure a suitable fill rate of the playback buffer. An example is given in the previous section in Figure 67 (b). The figure shows that due to saved data traffic over a long period of time, the buffer level can be kept at 100%, whereas the standard MPEG DASH adaptation schemes encounter stalling. Thus, the number and duration of stalls decrease, resulting in a reduced impact of stalling when VAS is used. This improvement can be observed for any adaptation scheme. The slightly worse performance of SQA in comparison to TQA is related to the risk of delaying decreasing adaptations, at the risk of additional stalling. For five of the seven adaptation schemes, no additional stalls using SQA could be observed. However, the average quality, including stalling events, is approximately 0.04 higher than in TQA. The effect of adaptations on the perceived quality is not considered, it is discussed in Section 7.5.2. In particular, the adaptation schemes reacting quickly to a throughput benefit most from applying VAS, i.e., hybrid and throughput-based adaptation schemes.

But independent of the adaptation scheme used, VAS improves the streaming experience by increasing the average quality of the streamed representations and stabilizing the buffer fill ratios which avoids stalling.

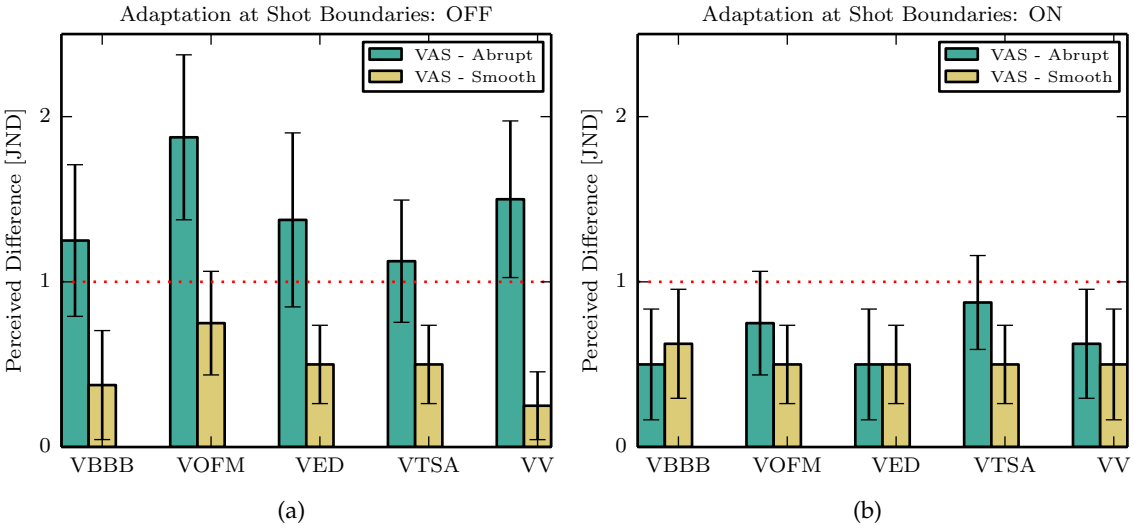



Figure 69: Perceived quality measured by the JND when using the VAS SQA ensuring a consistent quality placing the adaptations (a) at segment boundaries and (b) at video shot boundaries.

### 7.5.2 *Subjective Studies on the Impact of Adaptations*

The VAS service may introduce additional adaptations between different MPEG DASH representations. Table 26 indicates that the VAS SQA scheme achieves a slight increase in the objectively measured video quality. Note that the impact of an adaptation on the perceived quality is not measured by an objective video quality metric. In contrast to classical MPEG DASH adaptation schemes, VAS respects the content properties; also SQA is capable of leveraging intermediate representations when switching between quality levels.

As available quality metrics do not yet consider adaptation effects, subjective studies are conducted to analyze the real gains when using SQA. SQA is investigated in respect of the perceived quality when users watch an adaptive video. Adaptations at segment boundaries (HTTP GET requests) and at shot boundaries (partial HTTP GET requests) are compared in terms of their effect on human perception. The subjective study includes 23 test subjects watching selected video segments on a mobile device mediated by the crowdsourcing platform Crowdee[7]. The users' age ranged from 17 to 34. 18 of the assessors are male. The SQA smooth adaptation scheme is compared with the standard TB adaptation scheme. The maximum duration of each video segment used in this evaluation is a 30 seconds. It represents adaptations induced by the trace TM2 with a difference from the source to the target representation index of at least five index steps.

The differences in perceived quality are determined using the JND [**Watson2001**] in combination with a forced choice experiment, as described in Section 2.3.2. Adaptation assistance is evaluated by using three sequences: stable subjective quality, and the increase and decrease of quality. The video sequence selection is made based on varying SI, TI, and Co characteristics. Figure 69 shows the results of the subjective study in terms of the absolute JND. In a second step, the difference is determined for the perceived quality between a video streaming session that keeps a high bit rate representation. The analyzed situation is when a stable perceived quality results in a significant decrease of the MPEG DASH representation index. This gives an impression on how the impact of the adaptation

7 http://crowdee.de allows for conducting crowdsourcing studies on mobile devices; Visited on: 09/25/2016

is perceived. The experiment is conducted for all five video streams from which eight segments of at most 30 seconds are extracted. VAS performs MPEG DASH representation adaptations to ensure a stable perceived quality experience while minimizing data traffic. During the evaluation, it is ensured that no quality-degrading effects occur other than the adaptation (e.g., stalling or packet loss induced artifacts). Each subject has to evaluate 24 randomly selected combinations containing a reference video with a stable bit rate, and a video sequence using either SQA or TQA.

In addition, video adaptation schemes are placed either at MPEG DASH segment boundaries (see the results in Figure 69 a) or at the shot boundaries detected by VAS (see Figure 69 b). The depicted results give an idea on which adaptations are perceivable by humans. A JND of one is depicted as a red barrier when it is assumed that an adaptation is strongly perceivable by human observers on a mobile device.

For the case where adaptations are executed only after a complete MPEG DASH segment was played back (as evaluated in Figure 69 a), users notice considerable quality differences. It can be concluded that abrupt switches between a source and a target representation are observable on a mobile device. The VAS smooth adaptation scheme decreases the JND for the video sequences VBBB, VOFM, VED, and VV significantly. If the MPEG DASH client supports partial GET requests, adaptations should be planned to be executed at shot boundaries as this makes the adaptation transparent to the user. Executing adaptations at shot boundaries allows for adjusting the bit rate of the video stream with abrupt switches, which are nearly imperceivable for VBBB, VED, and VV. It can be concluded that VAS smooth adaptation ensures that for most evaluated videos, adaptations can be executed in a nearly imperceivable manner.

## 7.6 CONCLUSION

This chapter introduced VAS, a system to support video adaptation on mobile video streaming clients in a content- and quality-aware manner. The rationale behind VAS is that current MPEG DASH clients do not investigate the content encoded in a video. Especially in situations when a network has sufficient throughput to stream the highest representation, huge data savings are possible when VAS is used. VAS leverages image processing algorithms and an objective video quality metric to understand what is encoded within a video and how it is perceived by humans. VAS analyzes the video content on its structural, temporal and color characteristics to classify segments of videos with similar characteristics. It was shown that quality models can be mapped between segments of different video sequences without significant loss of precision. Both concepts allow real-time calculation of live video streams and scalability to a multitude of different video streams, which was demonstrated in a real deployment [**Wilk2016c**].

VAS allows clients to express the desired quality regarding the MOS and links it to MPEG DASH representations. Besides an optimal adaptation scheme, two adaptation support heuristics are introduced which support MPEG DASH adaptation schemes: TQA and SQA. The TQA allows to clients stream a specific target quality level, whereas classical adaptation schemes solely rely on bit rates. It aims for saving data traffic by not selecting MPEG DASH representations that surpass the desired quality. SQA adds an adaptation support method, which mitigates adaptation effects introduced by TQA and MPEG DASH adaptation schemes. SQA leverages the knowledge of the video content to execute adaptations in a covert manner. The evaluation indicates that significant data traffic savings can be achieved (up to 82.83%) without any decrease in quality.

# 8 CONCLUSION

This thesis discusses the advantages of integrating content adaptation into the design of networked video applications. Specifically, it discusses the scenario of an Mobile Video Broadcasting Service (MBS) that records video on smart mobile devices and uploads them using Internet Protocol (IP)-based network connections as a live stream. The scenario also includes the distribution of the generated live stream to viewers. Systems were designed and implemented that improve video adaptation significantly, and they were carefully evaluated. This chapter summarizes the thesis and shows its major contributions to the field of multimedia research. Finally, it gives an outlook on future research directions.

## 8.1 SUMMARY OF THE THESIS

The thesis aims at the design, realization, and evaluation of a live User-Generated Video (UGV) uploading, processing, and distribution system that leverages quality-aware content adaptation. Content adaptation is used at different steps of the video streaming process in the form of adaptive video streaming and video composition. Adaptive video streaming, as understood in this thesis, is the representation of a digital video in different quality versions, which differ regarding their average bit rate. During a streaming session, the video version is selected which suits best to the available throughput rates. By this, a video stream can be played back continuously without interruption.

In contrast, video composition assumes the availability of different video sources which record similar content as different video files. The result of a video composition is not a multitude of different video streams, but a single one consisting of selected video segments from each source. Thus, a video composition algorithm does not select a representation of a single video stream but which video to stream at any point in time.

Both content adaptation forms are used within this thesis in a way to enable *quality-aware* video streaming. The quality awareness depicts that the perceived utility for a user watching the stream should be improved when using content adaptation. The efficiency ensures that a video stream is produced at minimal costs for a given quality. This thesis uses the perceived quality determined by objective quality assessment algorithms as an indicator for quality awareness, and the generated data traffic as a metric for costs.

Chapter 1 introduces the scenario of a live upload of User-Generated Video (UGV), its processing, and delivery. It discusses challenges such as the unpredictable conditions of IP-based networks for a single device, the reduced quality of UGV productions, and the devices used to create the digital video streams. The chapter introduces the main goal of the thesis: *to design and realize a live UGV uploading, processing, and distribution system that leverages quality-aware content adaptation*.

Specific characteristics of digital video and its streaming are discussed in Chapter 2. Furthermore, it builds upon the scenario discussed in Chapter 1 by elaborating on how content adaptation can help to improve the quality of video streams. As UGV is different from professional content, there is a discussion about the perceived quality in UGV in a novel area, the so-called *recording degradations*. Whereas existing objective quality metrics inspect video for compression and transmission artifacts, UGV mainly suffers from poor recording conditions and human mistakes. During recording, degradations occur such as

camera shakes, objects occluding the view, and camera misalignments. In a discussion of related work, Chapter 2 shows that a research gap exists in this area. On the basis of a common understanding of quality, the chapter also introduces the Mobile Video Broadcasting Service (MBS), which aims at a live upload of videos. The chapter introduces different scenarios in which an MBS is used, and discusses the strength and weaknesses of existing systems. These systems usually lack the capability of content and mechanism adaptations, making them unreliable for efficient media provisioning. Video composition is introduced as the second form of content adaptation which is a novel understanding on quality in videos. Besides the technical and recording quality, composed videos gain a significant proportion of their quality from the influence of diversity and the observation of cinematographic rules. Existing work in this area is discussed to gain an understanding of the missing pieces provided by this thesis. Finally, Chapter 2 introduces background information and a discussion of existing work in the area of adaptive video delivery. It is shown that an adaptive video streaming approach usually neglects the content of a video and focuses on network conditions to make adaptation decisions. In summary, the chapter discusses the major differences of existing work and the novel contributions of this thesis.

In Chapter 3, new quality models are given for quantifying the impact of recording degradations. The most imminent degradations of camera shakes, harmful occlusions, and camera misalignments are discussed. Their impact on the perceived quality is quantified in a crowdsourced study with more than 1600 assessors. Besides the impact of degradations, the chapter discusses a study on the perception of different views of the same scene. The availability of such views is an essential requirement for video composition. The study shows that the recording position has a significant impact on the perceived quality and should thus be considered in the video composition.

These quality models are used for the design and development of the quality assessment framework Placement and Selection Component (PaSC), accompanied by novel algorithms. As introduced in Chapter 4, PaSC copes with the unscalable, centralized quality assessment by introducing a method to leverage the resources of arbitrary smart mobile devices and servers to conduct a *distributed quality assessment*. The PaSC achieves increased resource utilization by parallelizing quality assessment processing. As existing, efficient algorithms for the assessment of recording degradations are missing, the chapter introduces novel quality assessment algorithms that inspect recording degradations. To handle the tradeoff between the algorithm's high precision and a reduced runtime, *hybrid algorithms* are developed that not only consider time-intensive video analysis, but also inspect auxiliary sensors that record the context of a recording session. The proposed algorithms outperform existing work regarding the F1-score and the runtime, as well as offer a more fine-grained determination of video quality.

Chapter 5 describes how videos are uploaded, e.g., to a video composition server, in an efficient and content-adaptive manner. Live Video Upload System (LiViU) is introduced that is the first prototype for a live streaming application which supports the *parallel creation of video representations and ensures their live upload for further processing*. LiViU not only supports content adaptation, but also integrates the adjustment of the stream scheduling to cope with varying application and scenario requirements. The proposed system shows robustness against network condition changes, mobility, and copes with varying application requirements at runtime.

The findings and system components of Chapters 4 and 5 are used in the design and realization of a *novel video composition algorithm*. Video composition is one of two content adaptation concepts discussed in this thesis. It aims at creating superior quality composed video by selecting the best segments from a range of source videos. The videos are com-

posed on a central server in real-time, based on video streams provided by LiViU. In a first stage of the proposed composition algorithms, Chapter 6 describes the filter stage, ensuring that video views considered for composition have a minimal perceived quality using PaSC. Furthermore, the stage ensures that basic cinematographic rules are not broken when switching between segments of different video sources. As a result, a limited amount of video views is considered for composition in a second stage. The second stage either runs the semi-automatic CrowdCompose system or the real-time, automatic AutoCompose algorithm. CrowdCompose is based on the principle of crowdsourcing, which splits the complex task of video composition into several atomic tasks and mediates them to a group of anonymous users. Users compose the stream by conducting these tasks in parallel and create a superior quality video by leveraging human understanding of the scene. The composed videos and decisions are used to train the machine learning-based AutoCompose, which creates composed videos based on the decisions of CrowdCompose combined with a multi-feature analysis of a video stream. The resulting, composed videos have a lower quality in comparison to the manual video mixes but are superior to comparable video composition algorithms.

The composed video is transmitted to interested viewers. The delivery of the video stream leverages the second content adaptation principle, namely adaptive video streaming. Adaptive video streaming allows for leveraging different quality versions of a video to compensate for varying network conditions by switching between the versions. Existing work conducts this adaptation of video representations in a network-aware manner, but neglects considering the content of the transported video. The proposed Video Adaptation Service (VAS) allows a *content-aware video adaptation* by categorizing videos using Spatial Perceptual Information (SI), Temporal Perceptual Information (TI), and Color Perceptual Information (Co) descriptors. The features are linked to an objective quality metric to realize a quality lookup datasets, which allows for real-time adaptation of new video streams. As a result, not only can the perceived quality be kept, but also data traffic to mobile devices is reduced by up to 82.83%.

## 8.2 CONTRIBUTIONS

The contributions of this work start with fine-granular quality models for the degrading impairments of camera shakes, harmful occlusions, and camera misalignments. These models are the first, published quantification of the degradations' impact on human perception. A second contribution of Chapter 3 is the analysis of different recording positions and their influence on the perceived quality of, e.g., a composed video. Novel quality models are proposed which assess the quality of the recording position in relation to the distance and angle to a Point of Interest (PoI). These models are the basis for the contributions in the remaining chapters.

In addition, Chapter 4 leverages the models to design novel quality assessment algorithms for real-time detection of degradations. The proposed algorithms decrease the runtime by fusing visual and auxiliary sensor-based features while analyzing the videos. On several different UGV databases, it was shown that this hybrid analysis outperforms comparable work. Also, Chapter 4 discusses how to increase the scalability of quality assessment by introducing the PaSC. The PaSC allows for the distributed execution of quality assessment algorithms on servers and mobile devices to achieve increased scalability and reduced runtime of a complete assessment of a video stream.

Chapter 5 introduces the first prototypically evaluated MBS, which allows for adaptive video streaming and network mechanism adaptation. For content adaptation, the pro-

posed LiViU achieves an in-parallel creation of different video representations by leveraging hardware-accelerated transcoding on Android devices. Different representations can then be used to adapt the streamed video bit rate according to current network conditions. In combination with an adaptive scheduling of video stream messages, LiViU copes with changing network and application requirements. This allows LiViU to stream both to remote and close-by receivers. The proposed system achieves a reliable, in-time upload of live streams from mobile devices, despite mobility and varying network conditions.

Chapter 6 discusses contributions of the first semi-automatic, crowdsourcing-based video composition algorithm as well as an automatic composition application. By splitting and atomizing both the video composition task CrowdCompose shows how an improved quality video stream can be created. The resulting composed videos are leveraged to train the AutoCompose system, which achieves a real-time, automatic composition of UGV. It is the first composition algorithm that achieves a comparable quality to an amateur composition of a video for live streams.

The last contributions are discussed in Chapter 7, which introduces the quality-aware adaptive streaming system VAS. It offers the first content-aware adaptation system for Dynamic Adaptive Streaming over HTTP (DASH) clients that supports live streams. Also, it shows that the perceived quality models can be mapped between different videos as long as some sophisticated, yet easily calculated features exist for classifying video content. In the discussed version of the VAS spatial, temporal, and color features are discussed. As a result, the data traffic of streaming media can be reduced significantly.

Thus, in total, the thesis offers contributions towards the design, realization, and evaluation of a live UGV uploading, processing, and distribution system leveraging quality-aware content adaptation.

## 8.3 OUTLOOK

Based on the contributions of this thesis, further steps to improve content adaptation can be taken.

### 8.3.1 *Personalization and Distribution of Video Composition*

Existing video composition algorithms rely on the inspection of the video. In recent work by Stohr et al., an early-stage prototype towards the auxiliary data-based composition of videos is shown [**Stohr2016**]. A central element in this prototype is the idea of preprocessing the composition on mobile devices as proposed by the PaSC. The next step is a decentralized composition, when different mobile devices are coordinated to compose the video stream. This decentralization allows the placement of video composition functionality on mobile devices to support a composition in situ as well as in the fixed network, in which programmable network elements can support video composition as the media is transmitted through the network.

A second extension of our work considers the concept that a single video is composed of multiple streams. A composition is conducted to increase the perceived quality for an average viewer. As individual users can have different expectations of a video stream, a single composed video may not be enough. The personalization of the video composition can generate a multitude of composed videos, depending on user expectations and viewing habits. An assessment of how personalized videos should be composed as well as their efficient transmission over challenged networks, is still missing.

### 8.3.2 *Adaptive Upload of Video Streams*

The proposed LiViU system focuses on content adaptation and proposes adaptation of network mechanisms such as scheduling. Other mechanisms can also influence the perceived quality. Stohr et al. gave an interesting analysis on the effect of the congestion control mechanisms (e.g., in Transmission Control Protocol (TCP)) on the video streaming experience for distributing video [**Stohr2016b**]. It is shown that varying environmental conditions affect the performance of congestion control variants. Future MBSs may inspect the advantages of congestion control adaptations for video uploads.

### 8.3.3 *Context and Quality*

The perception of quality and its understanding are central concepts in this thesis. Still, little is known about the perception of video on mobile devices in different contextual situations, e.g., movement. These models are required as mobile video consumption grows tremendously. Ideal viewing conditions, as evaluated in current lab setups, are thus only of limited help to understand quality in mobile streaming sessions. The aspects of unstable viewing conditions have been integrated into the prototype of an environmentally-aware, video-streaming client [**Wilk2015b**]. Further research is needed to understand and leverage the context and environment when streaming video to mobile clients. This research can leverage the existing understanding of network conditions and content characteristics as offered by VAS.

### 8.3.4 *Information Centric Networks*

Novel paradigms are proposed to solve the existing challenges of the Internet by its complete redesign. Information-centric Network (ICN) is a novel concept which replaces host-centrality in the current Internet by information-centrality. Addressing of information or content becomes independent of the location of the data. The focus on information or content offers new opportunities for content adaptation for both adaptive video streaming and video composition. For example, the proposed VAS has to cope with some additional challenges as video segments can be distributed in an ICN on different servers. Moving Pictures Expert Group (MPEG) DASH leverages the throughput measurement of a video segment to select the the appropriate representation for the next segments. This can be dangerous if the segments are requested from different servers. An overview on upcoming challenges when combining ICNs and adaptive video streaming is presented by Wesphal et al. [**rfc7933**]. Video composition on the other hand benefits from a native support of content provider mobility in ICNs. Especially, when a large set of smart mobile devices produce live video streams, addressing and routing in ICNs promise to ease the selection of video streams for composition, as a virtual director sends a request for a desired video stream into the network. This request may include information such as the desired minimum quality and recording location. This would reduce the computational burden of video composition applications, which promises an increased scalability which is not available today.

# LIST OF FIGURES

# LIST OF TABLES

# LIST OF ACRONYMS

| | |
|---|---|
| **240p** | 320x240 |
| **360p** | 480x360 |
| **480p** | 1024x480 |
| **4CIF** | 704x576 |
| **576p** | 768x576 |
| **720p** | 1280x720 |
| **1080p** | 1920x1080 |
| **2160p** | 3840x2160 |
| **3G** | 3rd Generation Mobile Networks |
| **3GPP** | 3rd Generation Partnership Project |
| **AGBR** | Adaptive Guaranteed Bit Rate |
| **AMGS** | Automated Mashup Generation System |
| **ANSI** | American National Standard Institute |
| **AODV** | Ad-hoc On-demand Distance Vector |
| **AoI** | Area of Interest |
| **API** | Application Programming Interface |
| **AQ** | Audio Quality Assessment |
| **ARQ** | Automatic Repeat reQuest |
| **ASMA** | Adaptive Strategy for Mobile Ad-hoc streaming |
| **ATB** | Aggressive Threshold-based Adaptation |
| **AVC** | Advanced Video Coding |
| **B.A.T.M.A.N.** | Better Approach To Mobile Ad-hoc Networking |
| **BRISQUE** | Blind/Referenceless Image Spatial QUality Evaluator |
| **BSSID** | Basic Service Set Identification |
| **BTR** | Buffer-based Throughput Adaptation |
| **CBR** | Constant Bitrate |
| **CC** | Pearson Correlation Coefficient |
| **CDN** | Content Delivery Network |
| **CI** | Continuity Index |
| **Co** | Color Perceptual Information |
| **CPU** | Central Processing Unit |

| | |
|---|---|
| **CRC** | Cyclic Redundancy Check |
| **CUDA** | Compute Unified Device Architecture |
| **D2D** | Device-to-Device |
| **DASH-P** | HTTP POST-based DASH Upload |
| **DASH-U** | DASH Upload |
| **DASH** | Dynamic Adaptive Streaming over HTTP |
| **DCT** | Discrete Cosine Transform |
| **DMOS** | Differential Mean Opinion Score |
| **DRM** | Digital Rights Management |
| **FC** | Quality Feature Calculation |
| **FEC** | Forward Error Correction |
| **FIFO** | First-In First-Out |
| **FLOP** | Floating Point Operation |
| **FoV** | Field of View |
| **FPS** | Frames per Second |
| **FR** | Full Reference |
| **FS** | Filtering and Smoothing |
| **FSBMA** | Fast Subblock Block Matching Algorithm |
| **FSGPA** | Fast Subblock Gray Projection Algorithm |
| **GoP** | Group of Pictures |
| **GPA** | Gray Projection Algorithm |
| **GPGPU** | General Purpose Graphics Processing Unit |
| **GPS** | Global Positioning System |
| **GPU** | Graphics Processing Unit |
| **HAS** | HTTP Adaptive Streaming |
| **HD** | High Definition |
| **HEVC** | High Efficient Video Coding |
| **HLS** | HTTP Live Streaming |
| **HMM** | Hidden Markov Model |
| **HTML** | Hypertext Markup Language |
| **HTTP** | Hypertext Transfer Protocol |
| **HVS** | Human Visual System |
| **IBSS** | Independent Basic Service Set |
| **ICN** | Information-centric Network |

| | |
|---|---|
| **IEEE** | Institute of Electrical and Electronics Engineers |
| **IP** | Internet Protocol |
| **IPTV** | Internet Protocol Television |
| **ISO** | International Organization for Standardization |
| **ITU** | International Telecommunication Union |
| **JNA** | Java Native Access |
| **JND** | Just Noticeable Difference |
| **JRE** | Java Runtime Environment |
| **JSON** | JavaScript Object Notation |
| **LACES** | Live Authoring through Compositing and Editing of Streaming Video |
| **LDST** | Load/Store Operation |
| **LiViU** | Live Video Upload System |
| **LPC** | Luminance Projection Correlation |
| **LTE** | Long Term Evolution |
| **M** | Model Calculation |
| **MBS** | Mobile Video Broadcasting Service |
| **MFCC** | Mel Frequency Cepstral Coefficients |
| **MILP** | Mixed Integer Linear Programming |
| **MOS** | Mean Opinion Score |
| **MPD** | Media Presentation Description |
| **MPEG-TS** | MPEG transport stream |
| **MPEG** | Moving Pictures Expert Group |
| **MSE** | Mean Squared Error |
| **MSSIM** | Mean Structural Similarity Index |
| **MTV** | Music Television |
| **MVLC** | Multiple Video Layer enCoding |
| **NAL** | Network Abstraction Layer |
| **NAT** | Network Address Translation |
| **NR** | No Reference |
| **NTIA** | National Telecommunication and Information Administration |
| **NTP** | Network Time Protocol |
| **OLSR** | Optimized Link State Routing |
| **OpenGLES** | Open Graphics Library for Embedded Systems |
| **OS** | Operating System |

| | |
|---|---|
| **OSI** | Open Systems Interconnection Model |
| **OSMF** | Open Source Media Framework |
| **OSN** | Online Social Network |
| **P2P** | Peer-to-Peer |
| **PANDA** | Probe-AND-Adapt |
| **PaSC** | Placement and Selection Component |
| **PBS** | Personal Broadcasting Service |
| **PC** | Quality Parameter Calculation |
| **PCC** | Pearson Correlation Coefficient |
| **PEVQ** | Perceptual Evaluation of Video Quality |
| **PoI** | Point of Interest |
| **PSM** | Perceptual Sharpness Metric |
| **PSNR** | Peak Signal to Noise Ratio |
| **QDASH** | QoE-aware DASH |
| **QD** | QoE-aware DASH Adaptation |
| **QFAS** | Quality-Fair HTTP Adaptive Streaming |
| **QoE** | Quality of Experience |
| **RAM** | Random Access Memory |
| **REST** | Representational State Transfer |
| **RGB** | Red Green Blue |
| **RoI** | Region of Interest |
| **RQ** | Recording Quality Assessment |
| **RR** | Reduced Reference |
| **RT-VQM** | Real-Time Video Quality Assessment |
| **RTMFP** | Real-Time Media Flow Protocol |
| **RTMP** | Real-Time Messaging Protocol |
| **RTCP** | Real-Time Control Protocol |
| **RTP** | Real-Time Transport Protocol |
| **RTSP** | Real-Time Streaming Protocol |
| **S** | Sampling |
| **SAD** | Sum of Absolute Differences |
| **SARA** | Segment-Aware Rate Adaptation |
| **SDCF** | Sensor Data Collection Framework |
| **SfM** | Structure from Motion |

**SHVC** Scalable High Efficient Video Coding

**SI** Spatial Perceptual Information

**SIF** Source Interchange Format

**SIMD** Single Instruction Multiple Data

**SNR** Signal-to-Noise Ratio

**SQA** Smooth Quality Adaptation

**SROCC** Spearman Rank Order Correlation Coefficient

**SSCQS** Single Stimulus Continuous Quality Scale

**SSID** Service Set Identifier

**SSIM** Structural Similarity Index

**SSIS** Service Set Identifier

**ST-Region** Spatio-Temporal Region

**ST** Smoothed Throughput-based Adaptation

**stalling** freezing playback for rebuffering

**STB** Smoothed Throughput-based Adaptation

**SVC** Scalable Video Coding

**SVLC** Single Video Layer enCoding

**SVM** Support Vector Machine

**SVM-HMM** Support Vector Machine - Hidden Markov Model

**TB** Throughput-based Adaptation

**TBB** Threshold-based Buffer Adaptation

**TBST** Threshold- and Smoothed Throughput-based Adaptation

**TCP** Transmission Control Protocol

**TI** Temporal Perceptual Information

**TQA** Target Quality Adaptation

**TS** Transport Stream

**TV** Television

**UDP** User Datagram Protocol

**UDP-PL** UDP-Pull

**UGC** User-Generated Content

**UGV** User-Generated Video

**UI** User Interface

**UMTS** Universal Mobile Telecommunications System

**URL** Uniform Resource Locator

| | |
|---|---|
| **UTM** | Universal Transverse Mercator |
| **V-BLIINDS** | Video BLIINDS |
| **VAS** | Video Adaptation Service |
| **VMAF** | Video Quality Model with Variable Frame Delay |
| **VoD** | Video on Demand |
| **VQ** | Video Quality Assessment |
| **VQEG** | Video Quality Experts Group |
| **VQM** | Video Quality Metric |
| **VSS** | Video Sharing Site |
| **WAN** | Wide Area Network |
| **WLAN** | Wireless Local Area Network |
| **WSIF** | Wide Source Interchange Format |
| **WWM** | We want More! |

# A ADDITIONAL RESOURCES

This thesis introduces a range of novel systems, quality models, traces and datasets that are publicly available (at the time of the publication of this document).

The interested reader can find the resources at:
https://github.com/swilkTUDA/Dissertation_Resources.

# B

# AUTHOR'S PUBLICATIONS

## B.1 MAIN PUBLICATIONS

## B.2 CO-AUTHORED PUBLICATIONS

# C CURRICULUM VITÆ

## PERSONAL INFORMATION

| | |
|---|---|
| Name | Stefan Wilk |
| Date of Birth | January 28, 1987 |
| Place of Birth | Zweibrücken |
| Nationality | German |

## EDUCATION

04/2013 – 12/2016 Technische Universität Darmstadt
Doctoral candidate – Department of Computer Science

09/2009 – 08/2011 Universität Mannheim
Information Systems – Degree: Masters of Science

09/2006 – 08/2009 DHBW Mannheim
Information Systems – Degree: Bachelor of Science

## PROFESSIONAL EXPERIENCE

04/2013 – 12/2016 Technische Universität Darmstadt
Research assistant – Distributed Multimedia Systems (DMS)

09/2011 – 03/2013 PricewaterhouseCoopers WPG AG
Consultant CIO Advisory

09/2006 – 09/2009 Schaeffler KG
Student – Information Technology

## AWARDS AND HONORS

| | |
|---|---|
| since 2014 | SoftwareCampus Member: Educating IT Professionals |
| 2015 | Best Paper Award: **Schulz2015** |
| 2015 | Best Demo Award (3rd Place): **Wilk2015Demo** |
| 2014 | Best Demo Award: **Richerzhagen2014** |
| 2013 | Best Student Award: Erasmus Intensive Programme on Multimedia and the Future Internet: Moving Social and Mobile, Reading, UK |
| 2012 | Best Master Thesis Award: 'Stiftung Medien- und Kommunikationswissenschaften', Mannheim |
| 2011 | Best Master Thesis Award, Information Systems, Universität Mannheim |

## TEACHING ACTIVITIES

| | |
|---|---|
| Since 2013 | Technische Universität Darmstadt<br>Tutor for various Bachelor-, Master theses |
| Since 2013 | Technische Universität Darmstadt<br>Organizer for the Seminar Distributed Multimedia Systems |
| 2013 – 2015 | Technische Universität Darmstadt<br>Tutor for the Seminar Advanced Topics in Future Internet Research |

## SCIENTIFIC ACTIVITIES

| | |
|---|---|
| TPC | International Conference on Multimedia Systems, 2016 (Demo) |
| TPC | International Conference on Multimedia Systems, 2017 |
| Reviewer | Conference on Networked Systems (NetSys), 2015 |
| Reviewer | Multimedia System Journal (MMSJ), 2014 |
| Reviewer | Multimedia System Journal (MMSJ), 2015 |
| Reviewer | Multimedia System Journal (MMSJ), 2016 |
| Reviewer | Multimedia Tools and Applications (MTAP), 2016 |
| Reviewer | Transactions on Multimedia Computing, Communications, and Applications (ACM TOMM), 2015 |

# Erklärung laut §9 der Promotionsordnung

Ich versichere hiermit, dass ich die vorliegende Dissertation allein und nur unter Verwendung der angegebenen Literatur verfasst habe.

Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

*Darmstadt, 2016*

Stefan Wilk