RT-VQM: Real-Time Video Quality Assessment for Adaptive Video Streaming Using GPUs

Matthias Wichtlhuber
TU Darmstadt, Germany
mwichtlh@ps.tu-darmstadt.de

Gregor Wicklein
TU Darmstadt, Germany
wicklein@ps.tu-darmstadt.de

Stefan Wilk
TU Darmstadt, Germany
stefan.wilk@cs.tu-darmstadt.de

Wolfgang Effelsberg
TU Darmstadt, Germany
wolfgang.effelsberg@cs.tu-darmstadt.de

David Hausheer
TU Darmstadt, Germany
hausheer@ps.tu-darmstadt.de

ABSTRACT

Adaptive streaming systems gain rising relevance for streaming services. Therefore, the same video is offered in multiple quality versions to clients for adaptation during playback. However, optimizing adaptation in a Quality of Experience (QoE) centric way is difficult. Current systems maximize bit rate, ignoring that different types of adaptation (resolution, framerate, quantization) correlate differently and in a non-linear way with user's perception. User validated video quality metrics can provide precise quality information. However, measurements of state-of-the-art metrics show either high computational intensity or weak correlation with subjective tests. This makes large-scale offline quality assessment processing intensive while real-time constrained scenarios like live streaming and video conferencing are hardly supportable. Consequently, this work presents the Real-Time Video Quality Metric (RT-VQM), a real-time, Graphics Processing Unit (GPU) supported version of the widely used Video Quality Metric (VQM). RT-VQM introduces efficient filtering operations, hardware-supported scaling and high-performance feature pooling. The approach outperforms VQM by a factor of 30, thus enabling a real-time assessment of up to 9 parallel video stream representations up to High Definition (HD) 720 resolution at 30fps.

CCS Concepts

•**Information systems** → **Multimedia streaming;** •**Theory of computation** → *Massively parallel algorithms;*

Keywords

Adaptive Video Streaming; General Purpose GPU; Video Quality Metric; Real-time

MMSys'16, May 10-13, 2016, Klagenfurt, Austria

DOI: http://dx.doi.org/10.1145/2910017.2910600

1. INTRODUCTION

During the last decade, Internet-based video streaming has evolved from a niche market to a highly commercialized business sector which is predicted to grow even faster in the near future [8]. For payment-based business models such as Netflix [7], the delivery of video streams with a high QoE is highly important to ensure the loyalty of customers.

Modern streaming protocols such as Dynamic Adaptive Streaming over HTTP (DASH) [19] and layered video codecs such as the H.264/H.265 Scalable Video Coding (SVC) extensions [17, 10] aim at ensuring high QoE by adapting video quality (resolution, framerate, quantization) and thus bit rate to prevent freezing playback for rebuffering (stalling). However, quality adaptation and the increasing variety of end-devices lead to a bloat of different video quality versions to be chosen from. As an example, YouTube serves up to 20 parallel versions of the same video, while Netflix offers up to 120 different versions due to a combinatorial explosion of supported quality levels and video codecs [11]. The plethora of options renders optimal adaptation path selection difficult as bit rate and QoE is correlated in a different, usually non-linear way for each type of quality adaptation [27].

A quantification of visual quality impairments between a stream's different quality versions can be approximated by utilizing subjectively adjusted and validated Full Reference (FR) QoE metrics. FR metrics compare the different quality versions by extracting visual quality features and feeding the result into a validated human vision-based perception model [9]. However, models showing a high correlation to subjective user perception are computationally intensive leading to execution times several orders of magnitude higher than the assessed video's playback time (see Section 2.2 for measurements). Consequently, real-time quality assessment of live streams and adaptive video conferencing is not possible and offline quality assessment for Video on Demand (VoD) is resource consuming.

As a solution, this work presents the RT-VQM, a real-time, GPU supported version of the highly precise and standardized VQM of Pinson et al. [16]. Therefore, VQM is decomposed into its operations, analyzed and extended for optimized memory access, efficient feature extraction, hardware-supported interpolation methods and high-performance feature pooling algorithms. Motivated by the different types of scalability, we investigate VQM's reaction

to changes in resolution, framerate, and quantization, i.e., the Signal-to-Noise Ratio (SNR). RT-VQM's source code is contributed to the community for future research[1].

The remainder of this work is organized as follows: Section 2 describes background information on video quality adaptation and video quality metrics including execution time and correlation measurements. Section 3 provides an in-depth analysis of the VQM algorithm of Pinson et al. [16]. In Section 4 a GPU-optimized system design for a parallel version of the VQM algorithm is discussed. Performance/precision trade-offs and VQM's reaction to different types of quality adaptation are quantified in Section 5. Finally, Section 6 summarizes the results and provides an outlook on future work.

2. BACKGROUND AND RELATED WORK

The remainder part of this work focuses on H.264/H.265 scalable video codecs for the videos and is agnostic to the used video streaming protocol. We have chosen SVC based streaming [17, 10] to dynamically adapt the video at any point in time during a streaming session. Nevertheless, the system and results can be transferred to Constant Bitrate (CBR) HTTP Adaptive Streaming (HAS)-based systems such as MPEG-DASH without any loss of generality, as RT-VQM and the applied methodology is independent of the underlying video encoding or streaming protocol.

2.1 Scalable Video Encoding

Scalable video encodings offer the potential to adapt the video to changing network conditions at runtime. Well-established and standardized approaches include the scalable extensions of H.264 and H.265 [17, 10]. Both allow the decoding of a multi-layer bit stream enabling to switch between quality representations at run-time.

Thus, one video can be transmitted to multiple clients and each device can select the appropriate quality depending e.g. on the current network throughput. H.264 as well as H.265/SVC rely on a three-dimensional quality model including adaptation of the spatial, temporal dimension and quantization, i.e., SNR, dimension. Each combination of the quality dimensions is encoded in one video layer as depicted in Figure 1. Figure 1 shows an example of a scalable video cube, a common representation of SVC. An increase in one dimension requires preceding layers for decoding but also increases the quality of the respective video. The lowest representation is called the base layer and represents the only independently decodable layer. SVC enables seamless adaptation of video quality to network resources during playback.

2.2 Video Quality Metrics

This work enables video streaming systems to adapt according to the subjective perception of video quality in real-time. Thus, we investigate the potential of different Full Reference (FR) metrics for a usage in real-time settings and their correlation with human perception. FR metrics require both, an impaired and a reference video sequence. A summary of the findings is listed in Table 1, including execution time measurements obtained by comparing two 15s HD720 (1280×720) videos at 30fps using the hardware described beforehand. The video footage inspected is the well known, balanced LIVE Mobile Video database [13].

[1]https://github.com/mwichtlh/rt-vqm

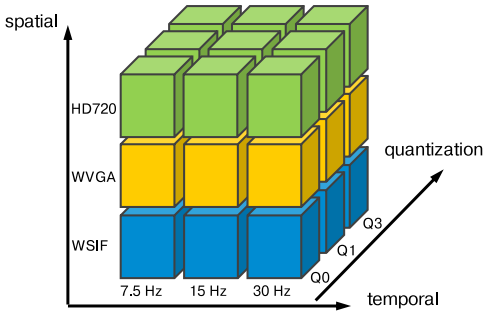


Figure 1: The H.264/H.265 SVC model represented as a cube of quality representations.

Simple and fast objective video quality metrics are approximating the perceived video quality by pixel comparison. These point-based metrics include the Peak Signal to Noise Ratio (PSNR), which measures the average error of pixel intensities between an impaired video sequence and its unimpaired reference. An advantage of PSNR and other similar point-based approaches is the low execution time of 1.6s. However, due to a Pearson Correlation Coefficient (CC) below 0.41 to the user's perception, PSNR only serves as a baseline for comparison.

VQM was developed by Pinson and Wolf for the National Telecommunication and Information Administration (NTIA) [26] and leverages quality calculation by extracting and comparing features in Spatio-Temporal Regions (ST-Regions) of a reference and an impaired video sequence. Contrast information is used for a majority of the influence factors of the VQM whereas other features leverage color information. During the VQEG Test Phase II, VQM outperformed several other state-of-the-art video quality metrics and was thus standardized [2, 3]. The execution time measurements of VQM show that the reference implementation is far from real-time capabilities (15s of video playback). However, of all metrics with a correlation higher than 0.6, VQM reaches the best ratio of execution time and CC.

A natural visual characteristics algorithm called Structural Similarity Index (SSIM) was introduced by Wang et al. [24]. SSIM leverages the assumption that especially changes in the structural information domain are important to describe perceived quality differences [23]. Thus, the authors defined a proportional decrease of perceived quality in relation to the decrease of structural similarity between an original and the distorted frame. In a subjective evaluation, SSIM shows a better performance than PSNR. A multi-scaled variant (MS-SSIM [25]) could even improve the accuracy. The main advantage of MS-SSIM is a dynamic adaptation of parameters according to the viewing conditions. In different viewing situations different scales show superior quality. However, since MS-SSIM was designed for still images, it does not consider the temporal dimension of a video. The authors of [22] extend MS-SSIM for a consideration of the temporal dimension. These extensions increase the correlation of SSIM on videos at the cost of computational complexity. Table 1 shows the measured execution time to be as high as 104s for a 15s video.

Seshadrinathan et al. developed the MOtion-based Video Integrity Evaluation index (MOVIE) [18]. Unlike SSIM, it does not directly examine video footage for visible distor-

Table 1: A comparison of state-of-the-art full-reference video quality metrics. Columns show the classification of the metric (Category), the support of temporal attributes (Motion), its execution time in seconds to compare two 15s of HD720 video at 30fps videos using the hardware described beforehand (lower is better), the correlation with subjective studies (CC, higher is better), a ratio of execution time and CC (lower is better) and a standardization status.

Algorithm	**Category**	**Motion**	**Exec. Time** [s] **Reference Impl.**	**Corr. (CC) LIVE DB** [20]	**Exec. Time per CC** [s]	**Standardization / Recommendation**
PSNR	point-based	agnostic	1.6	0.4035	3.97	no
VQM [26]	natural visual	supported	88	0.7236	121,61	ANSI/ITU [2, 3]
(MS-)SSIM [25]	natural visual	supported	104	0.7441	139.76	no
MOVIE [18]	perceptual	supported	2732	0.8116	3366.19	no
(ST-)MAD [12]	perceptual	supported	54876	0.8299	66686.11	no

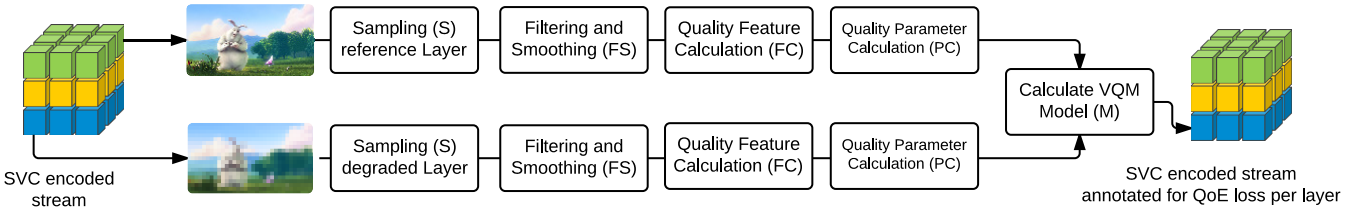


Figure 2: Essential steps to perform a VQM measurement (inspired by [16]). The video frame is taken from the Big Buck Bunny movie (http://www.bigbuckbunny.org), (c) Blender Foundation.

tions such as the loss of structures. Instead, MOVIE tries to imitate the perception of the Human Visual System (HVS) and is thus classified as a perceptual video quality metric. MOVIE achieves similar results or even outperforms existing, leading metrics in terms of correlation to user's perception (see Table 1). The disadvantage of MOVIE is the resulting processing time caused by a complex Gabor decomposition [14]. MOVIE requires more than 2700s for calculating the quality of an impaired 15s video. It has been shown that calculations including Gabor decomposition can be efficiently implemented on a GPU with an acceleration of more than 30 times [21]. However a real-time calculation would require a speedup by a factor of 2000.

Larson and Chandler developed Most Apparent Distortion (MAD) an algorithm for quality assessment in images [12]. MAD mimics the HVS perception of different levels of distortions in images and shows a better correlation with the HVS than PSNR and SSIM. They apply luminance and chrominance filters to create a visibility map. This map indicates if distortions are present in different areas of the image. Since MAD was designed for quality assessment in images, Vu et. al. proposed spatio-temporal MAD (ST-MAD) to apply the metric to video footage [20]. Even though Phan et. al. utilized GPU acceleration to accelerate ST-MAD by a factor of 47, the algorithm is still far from real-time capabilities [15]. The measured execution time for ST-MAD is as high as 15 hours for 15s of video.

The ITU-R J.247 describes Perceptual Evaluation of Video Quality (PEVQ) [4] as the leading perceptual FR video quality metric. PEVQ is a very robust approach as it relies on a very small set of features. It relies on the analysis of temporal, spatial, chrominance and luminance distortions between a reference and a test sequence. The algorithms relies on a five stage process. An initial stage excludes the border regions as distortions in this area are usually not perceived as degrading the quality. In a second step temporal misalignments between the reference and test sequence are detected and corrected. During the processing step the luminance values are adjusted. The third step includes a spatial alignment and the correction of the chroma components. During the last two steps first the distortion analysis is performed based on the previously mentioned features and mapped into a single Mean Opinion Score (MOS) value. However, the source code is patented and not freely available for evaluation. Due to its complexity the algorithm should be significantly slower than VQM and harder to be parallelized. Thus, PEVQ has been excluded from our evaluation and as a candidate for real-time video quality evaluation.

The synopsis presented in Table 1 reveals that none of the discussed metrics seems to be appropriate for real-time and highly correlated quality assessment, i.e., less than 15s of execution time and a higher CC than 0.6 when comparing the metric's output to subjective tests. Especially the perceptual based metrics MOVIE and ST-MAD stand out with a high CC but pay for the increased precision with an exponentially higher execution time. MS-SSIM and VQM offer a good trade-off of execution time and CC. MS-SSIM reaches a slightly higher CC than VQM, while VQM offers a lower execution time and thus a better ratio of execution time per CC. Consequently, VQM is chosen as a base for this work also taking into account that VQM is validated in large scale user studies [16] and standardized by the American National Standard Institute (ANSI)/International Telecommunication Union (ITU).

3. VQM ANALYSIS

The fundamental steps of VQM to achieve a highly correlated approximation of video quality for SVC streams are depicted in Figure 2. The assessment starts with a pairwise extraction of the different SVC representations from the encoded stream, where the highest quality layer acts as a reference for all other layers. Both videos are decompressed to the YCbCr color space (one luma channel Y and

two chrominance channels CbCr). In a Sampling (S) step, the data is converted to a floating point representation to provide enough precision for the following calculation steps.

Afterwards, the Filtering and Smoothing (FS) step applies an edge filter to the frames and calculates a set of measures for edges and a measure for the temporal changes. The Quality Feature Calculation (FC) step pools the output of the FS step to a set of single feature values per ST-Region, i.e., a disjoint cube of pixels in the horizontal and vertical dimension spanning 0.2s of video (e.g., 8×8 pixels spanning 6 frames at 30fps). Subsequently, the ST-Regions from both video sequences are compared by the Quality Parameter Calculation (PC) step, resulting in an ST-Region error matrix quantifying the feature-wise difference between both videos. The errors for each feature set are pooled spatially and temporally to a single value per feature. Finally, the single value representations for each feature are combined in the VQM Model Calculation (M) step by calculating a user-validated linear regression model. The resulting impairment value is annotated to the respective SVC layer to quantify the QoE loss compared to the highest layer.

The decomposition of VQM into its single processing steps allows for an assessment of its potential to be accelerated. For that purpose, three metrics are measured for each step of the reference implementation of VQM: the *absolute throughput* T, the *arithmetic intensity* A and the *absolute execution time* E.

The absolute throughput is defined as:

$$T\,{}^{\mathrm{MPixel}}/_{\mathrm{s}} = \frac{r_h * r_v * |f|}{t}, \tag{1}$$

where r_h and r_v are the horizontal and vertical resolution, $|f|$ represents the number of frames and t is the length of the assessed scene. Intuitively, T measures the number of pixels of a video sequence that can be processed per second. Consequently, if the sum of all encoded layer's pixels per second in an SVC stream is smaller than T, the whole SVC cube can be assessed in real-time.

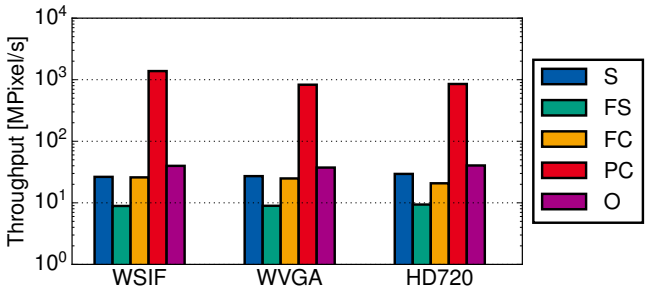
T is a well suited measure for identifying bottleneck processing steps worth to be accelerated. Nevertheless, T can not indicate the structural suitability of a processing step for parallelization. Therefore, the arithmetic intensity of each step is measured.

Arithmetic intensity is defined as:

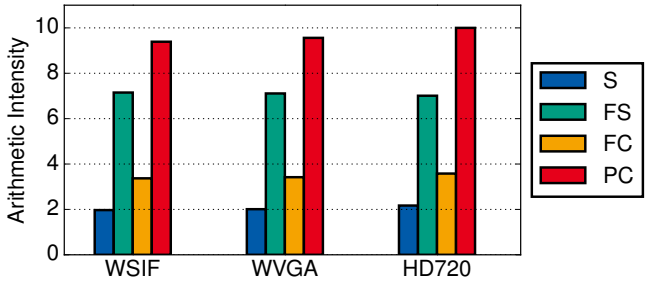
$$A\,{}^{\mathrm{FLOP}}/_{\mathrm{LDST}} = \frac{C\,\mathrm{FLOP}}{M\,\mathrm{LDST}}, \tag{2}$$

where C represents the number of Floating Point Operations (FLOPs) and M is the number of Load/Store Operations (LDSTs), i.e., memory access operations. Intuitively, A defines the fraction of clock cycles spent performing calculations compared to the cycles spent for accessing and storing data in memory. A high arithmetic intensity identifies steps bound by processing capabilities and not bound by Input-/Output (I/O) operations, which makes them especially suitable for a parallelization using GPUs [6].

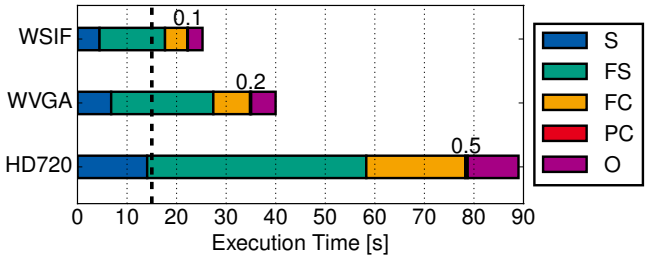
The measured results of the reference implementation are depicted in Figure 3. The hardware used for all measurements presented in this work is based on an Intel Core i7 4770 Central Processing Unit (CPU) running at 3.4GHz. The machine's Random Access Memory (RAM) is as large as 16GB. Additionally, a GeForce GTX 780 Ti GPU is



(a) Single step absolute throughput T; category Other (O) represents parts of the code that could not be assigned clearly.



(b) Arithmetic intensity A; only Level 1 cache misses considered.



(c) Absolute execution time E; category Other (O) represents parts of the code that could not be assigned clearly. The dashed line represents the performance necessary to achieve real time operation for the comparison of *two* versions of the stream, only.

Figure 3: Metric distribution of reference VQM version over processing steps. Results are averaged over 10 runs of a 15s video sequence from the LIVE Mobile database [13] at 30fps. The standard deviation is smaller than 1% for all results.

used to measure the GPU code. The GPU runs at 1.05GHz and can access 3GB of RAM. Both are connected via a Peripheral Component Interconnect (PCI) express bus with 16 lanes.

For obtaining the measurements, two video sequences of 15s at 30 fps are compared using three different resolutions covering HD720 (1280×720), WVGA (720×400) and WSIF (680×384). All measurements are averaged over 10 runs, the standard deviation was always smaller than 1% of the average value.

The results in Figure 3a show that the overall throughput is limited by the Sampling (S), Filtering and Smoothing (FS), and Quality Feature Calculation (FC) steps. Moreover, the reference implementation executes a large fraction of time intense copy operations that cannot be assigned clearly to one of the steps (category Other (O)). The Quality Parameter Calculation (PC) step is outstanding showing a comparably high throughput caused by the very small ab-

solute execution time (Figure 3c) of this step. Notably, the throughput stays remarkably constant when comparing the three resolutions, thus indicating the performance to scale linearly with the number of pixels to be processed.

Figure 3b depicts the arithmetic intensity per step. In order to account for the caching capabilities of modern CPU hardware, only Level 1 cache misses, i.e., memory access operations that cannot be served from the on-chip cache, are counted as LDST operations. Remarkably, the Quality Parameter Calculation (PC) calculation step has the highest A of all steps. However, the absolute runtime of this step (Figure 3c) indicates that Quality Parameter Calculation (PC) is not critical for real-time assessment, as the step accounts for less than 1% of the overall run time. As expected for classical graphics processing algorithms, the Filtering and Smoothing (FS) and Quality Feature Calculation (FC) step have a comparably high arithmetic intensity indicating they are well suited for a parallelization on a GPU. Finally, the Sampling (S) step has the lowest arithmetic intensity indicating it should be the last of the sub-algorithms to be parallelized.

4. RT-VQM DESIGN

In the preceding chapter, steps suitable for a massive parallelization of the VQM algorithm are identified, laying the base for deciding on which parts of the algorithm should be running as a kernel function, i.e., an independently executable unit of code executed by the GPU. However, besides the three criteria A, T, and E, the restrictions of bus transfers have to be taken into account. In fact, the GPU constitutes a separate co-processing unit. Thus, it behaves like a separate device running parallel to the CPU (host) and possesses its own memory. Consequently, all data to be processed by the GPU has to be transferred using the PCI bus, which can - despite the large theoretical throughput – easily become a performance bottleneck. Thus, RT-VQM's design presented in this chapter is not only based on the three metrics, but also chosen in a way that minimizes bus transfers. Based on these architectural considerations and our measurements, the following design decisions for RT-VQM are taken.

For the Sampling (S) step, the low absolute throughput at a low arithmetic complexity is a border case for a parallelization. Low arithmetic complexity suggests that an optimized, non-parallel implementation might be able to solve the sampling problem efficiently as well. Nevertheless, sampling creates a considerable bloat of video data resulting in a bus bottleneck. In particular, sampling increases the bit depth of a pixel from 12bits to 192bits, i.e., the video data volume is increased by a factor of 16 due to a conversion to a double precision floating point representation and a reorganization of the Cb and Cr channel.

We account for the data bloat during sampling by two countermeasures: first, the sampling step is handled by the GPU. Thus, the data bloat is happening in the GPU memory after the bus transfer of the unsampled video data. Second, the video data is sliced before transferring the data to the GPU's memory as this allows to already process video data on the GPU while transferring the next slice from the host's memory. This optimization allows to nearly completely hide the bus latency by interleaving processing and transfer of the data (asynchronous transfers/bus latency hiding). However, an arbitrary slicing is not possible, as the FS

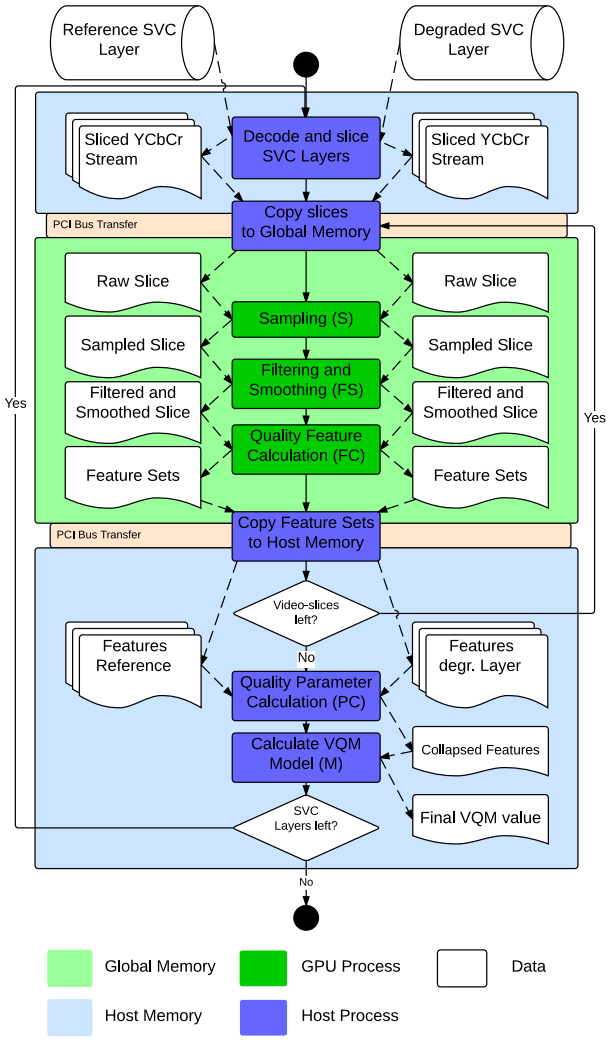
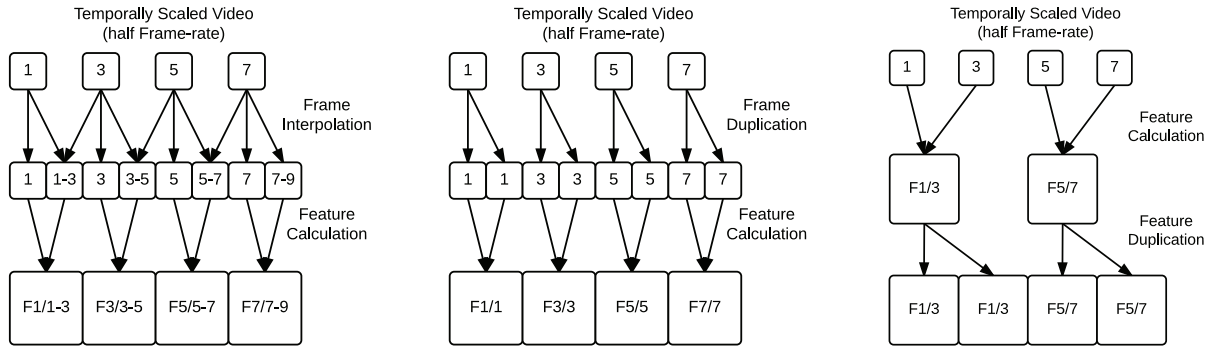


Figure 4: Proposed VQM calculation process on a GPU system. A detailed overview of the FS and FC steps is provided in Figure 6

and FC steps rely on complete ST-Regions to calculate features. Consequently, the slicing step cuts the video according to the borders of ST-Regions in the temporal dimension, i.e., usually a number of six unsampled consecutive frames forming one complete grid of ST-Regions is transferred at the same time.

Moreover, sampling on the GPU allows to account for the different types of scalability induced by SVC in a fast and elegant way. As VQM is defined and validated for comparing video material of equal resolutions and frame rates, a fast interpolation method is necessary before the comparison is done. For that purpose, the texture units of the GPU are utilized by treating the video material as a three dimensional texture. Three dimensional textures enable linear interpolation in hardware along three dimensions, i.e., accessing an interpolated texture value can be performed as fast as accessing a memory location. Using the texture units allows to treat the video frames like a non-discrete field of interpolated values generated from the supporting grid provided by the video material. Consequently, the sampling step can



(a) Frame Interpolation mode (b) Frame Duplication mode (c) Feature Duplication mode

Figure 5: Description of designed and evaluated temporal interpolation methods.

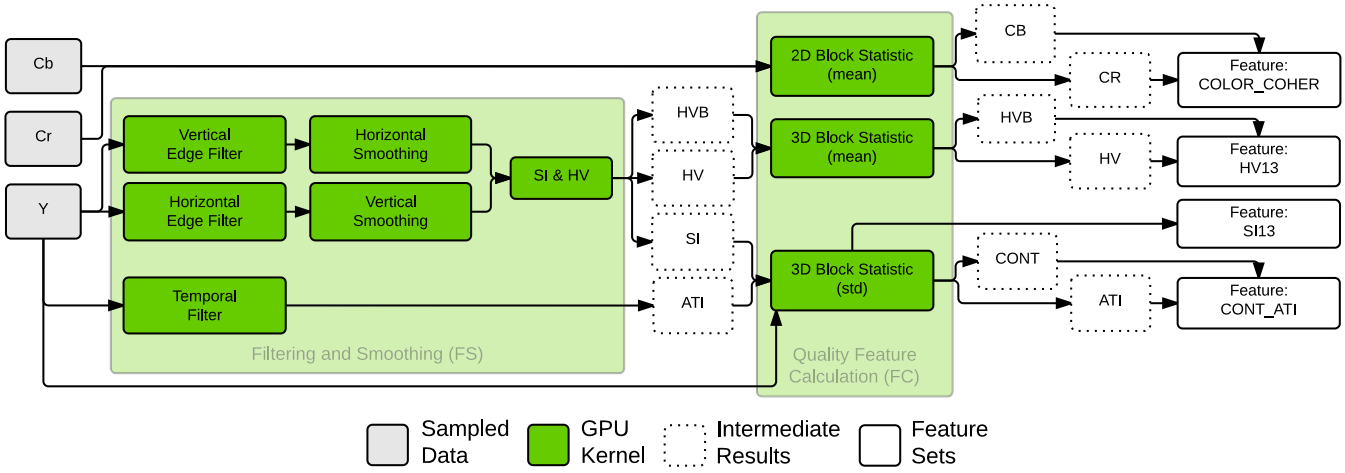


Figure 6: Close-up of the GPU based Filtering and Smoothing (FS) and Quality Feature Calculation (FC) process and intermediate results. The green boxes show the splitting of processing steps of VQM to GPU kernel functions.

provide a fast spatial and temporal upscaling to the reference video's resolution and framerate.

However, the upscaling along the temporal dimension leaves further room for optimization. It might also be possible to double the frames or even the feature sets without large impact to VQM's precision. The latter can be expected to be faster by an order of magnitude. Thus, three different modes are implemented: the Frame Interpolation mode (see Figure 5a) interpolates missing frames as described beforehand. This mode has the advantage of being able to handle temporal downscaling even with frame rates other than $\frac{1}{2^n}, n \in \mathbb{N}$ of the reference video's frame rates. For all other cases, the Frame Duplication mode (Figure 5b) and Feature Duplication mode (Figure 5c) may be more advantageous in terms of execution time. The impact on precision is evaluated in Section 5.

After sampling, the Filtering and Smoothing (FS) and Quality Feature Calculation (FC) steps are considered. Both are classical GPU operations. In particular, the Filtering and Smoothing (FS) step has a high arithmetic intensity at low throughput, offering large potential for parallelization. The operations of both steps are restructured and divided to run on nine different GPU kernel methods (see Figure 6).

The FS step is split into a *spatial filtering* process and a *temporal filtering* process. The spatial filtering process depends on one horizontal and one vertical filtering operation using a convolution matrix of size $K = 13 \times 13$ on the luminance channel. As for Sobel filtering, the operation can be decomposed into two operations using two convolution matrices $K = K_e \cdot K_s$, where K_e performs edge detection and K_s performs smoothing, and $\cdot$ denotes a matrix multiplication. RT-VQM utilizes this optimization and splits the Filtering and Smoothing (FS) to four GPU kernel functions, i.e., two edge filters and two smoothing filters (see Figure 6). This decomposition allows to run four compact kernel functions that can be executed with a higher hardware utilization than one large kernel function, as all necessary values to perform the convolution can be kept in the GPU's limited on-chip memory. As opposed to that, calculating the full 13×13 kernel would require storing intermediate results in the GPU's comparably slow global memory.

After edge filtering and smoothing in horizontal as well as vertical direction, the results are combined by the SI&HV kernel. Basically, the SI&HV kernel distinguishes the gradients calculated by the preceding kernels based on two metrics. The first is the length of the gradient:

$$R(i,j,t) = \sqrt{H(i,j,t)^2 + V(i,j,t)^2}, \tag{3}$$

where $H(\cdot)$ denotes the horizontally filtered frame and

$V(\cdot)$ denotes the vertically filtered frame. The second is an approximation of the direction of the gradient:

$$\theta(i,j,t) = \tan^{-1}\left[\frac{H(i,j,t)}{V(i,j,t)}\right] \approx \frac{\min(H(i,j,t), V(i,j,t))}{\max(H(i,j,t), V(i,j,t))}. \quad (4)$$

Both values, R as well as θ are calculated by the SI&HV kernel and are used to extract three features: The value of R is integrated into the SI set. The value of θ is used to separate detected edges into vertical and horizontal edges indicating block artifacts (HV) and diagonal edges (HVB), given R exceeds a threshold $R_{\min}$.

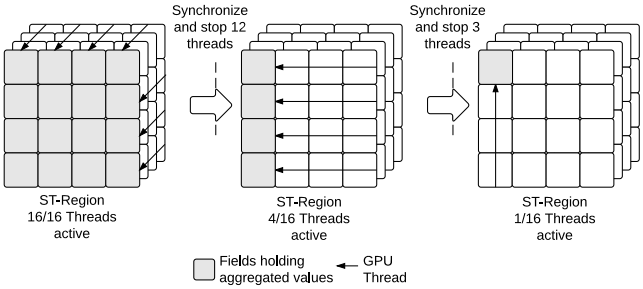
These operations impose several challenges for a parallel implementation resulting in the SI&HV kernel being one of the most complex of all kernels designed in the scope of this work. First filtering the values for a threshold $R_{\min}$ induces branching into different paths of execution. Branch divergence is problematic as GPU processors' design follows the Single Instruction Multiple Threads (SIMT) principle, i.e., usually the same instruction is executed on an array of data. However, if branching based on input data occurs, the two execution branches are executed sequentially which results in a low hardware utilization. Moreover, the calculation of R includes a square root operation which is computationally intensive. However, splitting this kernel into two smaller ones to avoid branch divergence is a bad solution, as this requires storing and reloading of R values using the GPU's slow global memory. The final design of the SI&HV kernel is a compromise between branch divergence, kernel complexity and global memory transfers. Nevertheless, the kernel might be optimized in the future when GPUs with a higher I/O performance are available.

As opposed to spatial filtering, the temporal filtering logic is uncritical and can be realized as a single kernel, as it merely calculates the difference in luminance between consecutive frames. The results (ATI) are written to global memory on a per-frame basis.

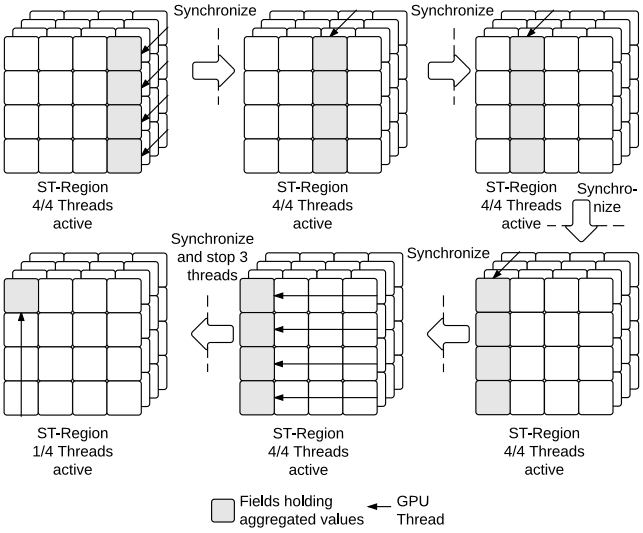
After the FS step, the four intermediate results are pooled to a single feature value per ST-Region by the Quality Feature Calculation (FC) step. The measurements of the reference implementation depicted in Figure 3 indicate that the FC step has a slightly higher throughput T than the FS step, however the arithmetic intensity is only roughly half as high. This step is again a border case for parallelization. The decision to perform this step on the GPU is again based on considering PCI bus transfer times: after pooling the feature per ST-Region the data to be considered for a transfer over the PCI bus is much smaller than before, i.e., only one value per ST-Region has to be transferred per feature (HVB, HV, SI, ATI). More precisely, the pooled data volume was measured to be only as high as 3% compared to the non-pooled data volume, thus rendering bus transfer times negligible.

In order to perform the FC step, optimized kernels for different statistical operations (mean and standard deviation) on two dimensional (per-frame) and three dimensional (per ST-Region) data structures are utilized as depicted in Figure 6. Nevertheless, pooling data with a large number of threads on a GPU is challenging.

The main problem is to ensure a high hardware utilization: GPUs are only capable to run batches of threads with a fixed size W, where W can be parameterized before



(a) Naive pooling of a 4x4x4 ST-Region with batch size $W = 16$. The batch utilizes 44% of the reserved resources.



(b) Optimized pooling of a 4x4x4 ST-Region with batch size $W = 4$ threads. The batch utilizes 87.5% of the reserved resources.

Figure 7: Optimized feature pooling versus naive pooling.

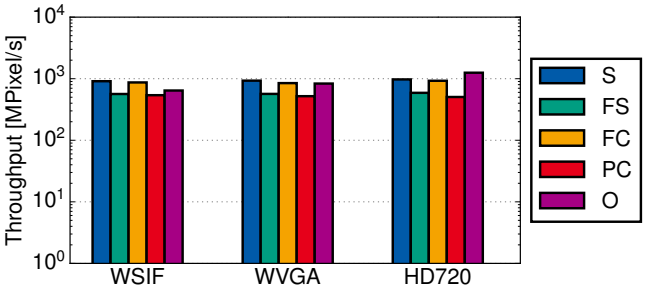
starting the kernel but stays constant throughout execution. However, the nature of a pooling procedure requires a single value as an outcome requiring exclusive memory access whenever threads running in parallel aggregate values across the threads. This type of access synchronization leads to a serialization of threads resulting in low hardware utilization.

The need to provide an optimized pooling method becomes evident when comparing a naive pooling method (Figure 7a) with the proposed and implemented optimized pooling method depicted in Figure 7b. The naive pooling method uses a batch size of $W = 16$ for collapsing in three steps along the three dimensions. Only the first step utilizes the full batch of 16 threads. In the second step, 12 of 16 reserved threads are idle and in the third step 15 of 16 reserved threads are idle, resulting in an average utilization of 44% of the reserved resources per step. As opposed to that, the optimized pooling method only uses a batch size of $W = 4$ threads, but uses all four threads during the pooling process except for the last step, where 3 threads are stopped and one thread performs the final aggregation. Consequently, an average utilization of the reserved resources of 87.5% per step can be achieved. Notably, the gap increases in a cubic way when increasing the size of the ST-Region. As opposed to Figure 7, most of VQM's features rely on larger ST-Regions of size $8 \times 8 \times 6$ [26, 16].

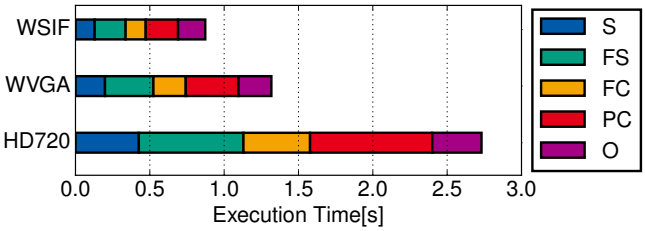
After the Quality Feature Calculation (FC) step, six feature sets (CB, CR, HVB, HV, CONT, ATI) reside in the GPU's global memory (see Figure 6). VQM defines four aggregated features (COLOR_COHER mapping color changes, HV13 mapping edge changes, SI13 mapping structural changes, and CONT_ATI mapping temporal changes weighted by contrast changes) as cross products of the single features. The aggregated features will not be explained in detail in this work, a more precise description can be found in [26, 16]. Notably, all feature values calculated up to this point exist per ST-Region.

Calculating one value per feature for the entire sequence is done in the Quality Parameter Calculation (PC) step, which performs a second pooling step over all previously pooled ST-Regions. The performance analysis of VQM revealed a high throughput at a low absolute runtime of about 0.5s for the PC step (see Figure 3). The reason for the comparably high throughput of the reference implementation is the low data volume to be processed, as the preceding FC step condenses the data volume to be processed by 97%. Thus, the data volume is small enough to be transferred back efficiently using the PCI bus. At the same time, there is no promising potential for a parallelization of the PC step. Consequently, RT-VQM switches back to the CPU for all following processing steps before executing the PC step (see Figure 4).

The outputs of the PC step are seven parameters for the linear regression model calculated in the final VQM Model Calculation (M) step. For the most precise regression model defined by VQM [26, 16], this requires an addition of seven weighted values, which is negligible in terms of execution time. In fact this step is performed so fast, that the measurements of the reference implementation in Section 3 were omitted due to insignificance compared to the other steps. Consequently, this last step remains on the CPU.



(a) Single step absolute throughput T; category Other (O) represents parts of the code that could not be assigned clearly.



(b) Absolute execution time E; category Other (O) represents parts of the code that could not be assigned clearly. Run time required for real-time processing is 15s.

Figure 8: Metric distribution of RT-VQM over processing steps. Results are averaged over 10 runs of a 15s video sequence at 30fps. The standard deviation is smaller than 1% for all results.

5. EVALUATION

The evaluation is driven by three research questions: first, the adherence of RT-VQM to the non-parallel reference version of VQM is investigated with respect to precision. In particular, we quantify the deviation from the reference implementation and investigate the trade-off between performance and precision. Second, the performance gain in terms of throughput and execution time as defined in Section 3 is measured. Special interest is paid to the question how many SVC layers can be assessed in real-time. Moreover, we investigate the suitability of the VQM general model with respect to the scalability modes of SVC and the hardware-supported linear interpolation methods along the spatial and temporal dimension introduced in Section 4.

5.1 Adherence to Reference

First, the adherence of RT-VQM to the reference implementation is validated. Therefore, the quality measurement deviations of the reference VQM version and our approach are compared. For the final M step, seven parameters are used to calculate a linear regression model. We list the mean, standard deviation and the maximum of the deviations compared to the reference in Table 2. The values are calculated using three videos from the LIVE Mobile database [13] and comparing them to 16 impaired versions from the same database for three different resolutions. Thus, the results are based on 144 video comparisons.

The reference VQM version operates with double precision floating point representations only. As GPUs are optimized for single precision calculations and tend to run considerably faster in this mode of operation, we have implemented the possibility to switch the RT-VQM implementation to a single precision mode. Thus, all floating point operations and the storing of all intermediate values is handled in single precision. This allows quantifying the loss of precision between both modes, i.e., the trade-off of precision and performance.

Notably, RT-VQM produces only slight errors in both modes. In double precision mode, the maximum deviation from the reference is in the range 2.78×10^{-16} for the HV_LOSS feature. Why this mode shows any difference at all, could not be clarified entirely. The authors assume that slight differences in the hardware implementation of the IEEE 754 standard [1] and different numerical algorithms for certain operations (e.g., square roots) could be a cause. Compared to double precision mode, the deviations in the single precision mode are higher. Again, the highest deviation can be observed for the HV_LOSS feature. The maximum total loss of the complete regression model is as high as $4.44 \times 10^{-16}/2.19 \times 10^{-6}$ for both modes.

Summing up, the deviations in both modes are negligible, as they are below 0.1%. Given that VQM quantifies human visual perception, which is imprecise by nature, it is highly unlikely that the identified deviations are measurable at all in a subjective test. Thus, for our objectively estimated quality of video we accept those negligible deviations and leverage a single precision mode-based calculation of VQM.

5.2 Runtime Performance Evaluation

As for the reference implementation, a performance evaluation is conducted using the throughput (T) and the abso-

Table 2: Precision of the VQM feature values compared against results from the reference implementation for single and double precision calculations. For a discussion of the regression model and the involved feature parameters see [16, 26].

Double Precision	**si_loss**	**hv_loss**	**hv_gain**	**color1**	**si_gain**	**contati**	**color2**	**total**
Mean(error)	2.86×10^{-17}	2.32×10^{-17}	2.13×10^{-17}	0	3.11×10^{-17}	-3.14×10^{-19}	0	9.44×10^{-17}
Std(error)	1.71×10^{-17}	9.47×10^{-17}	2.18×10^{-17}	0	1.24×10^{-16}	8.42×10^{-19}	0	1.55×10^{-16}
Max(error)	6.94×10^{-17}	-2.78×10^{-16}	8.33×10^{-17}	0	3.36×10^{-16}	-1.95×10^{-18}	0	4.44×10^{-16}
Single Precision	**si_loss**	**hv_loss**	**hv_gain**	**color1**	**si_gain**	**contati**	**color2**	**total**
Mean(error)	-5.59×10^{-9}	1.64×10^{-7}	-5.14×10^{-8}	3.56×10^{-10}	-1.55×10^{-8}	1.00×10^{-10}	1.35×10^{-9}	9.35×10^{-8}
Std(error)	2.17×10^{-8}	6.25×10^{-7}	1.02×10^{-7}	3.78×10^{-8}	9.49×10^{-8}	1.28×10^{-9}	2.11×10^{-9}	6.87×10^{-7}
Max(error)	-5.23×10^{-8}	2.27×10^{-6}	-2.50×10^{-7}	1.59×10^{-7}	2.24×10^{-7}	3.39×10^{-9}	6.69×10^{-9}	2.19×10^{-6}

Table 3: Comparison of total execution time E and total throughput T for reference VQM implementation and RT-VQM. The measurements were conducted on 15s video sequences at 30fps.

	WSIF		**WVGA**		**HD720**	
	VQM	**RT-VQM**	**VQM**	**RT-VQM**	**VQM**	**RT-VQM**
Mean (E) [s]	25.14	0.92	39.76	1.36	88.50	2.70
Speedup	27.33		29.23		31.72	
Speedup (rel.)	6.83		7.31		7.93	
$T[^{\text{MPixel}}/_{\text{s}}]$	4.66	127.72	4.63	135.64	4.69	153.6

lute execution time (E) as defined in Section 3. The results are depicted in Figure 8.

Notably, the throughput results are more equally distributed across the different processing steps than for the reference version (see Figure 8a) indicating no clear processing bottleneck anymore. Moreover, the absolute throughput of all steps is now around $10^3{}^{\text{MPixel}}/_{\text{s}}$. As opposed to that the steps in the reference version only reached a throughput between $9^{\text{MPixel}}/_{\text{s}}$ and $29^{\text{MPixel}}/_{\text{s}}$ except for the PC step.

The throughput results are reflected in the absolute execution time measurements (see Figure 8a): the distribution of time spent on processing the different steps is more uniform than for the reference version. Moreover, the PC step consumes a larger fraction compared to the other processing steps. In total, the absolute execution time for processing a video sequence in HD720 resolution does not consume more than 2.7s, which is far below the real time border of 15s playback time for the whole sequence.

A direct comparison of the total execution time and the reference implementation is listed in Table 3. For that purpose, the mean runtime for both versions is compared. For three different resolutions, a speedup of RT-VQMranging from a factor of 27 to 30 can be observed compared to the reference version. As the reference version is running single-threaded, we define a relative speedup by dividing the runtime of the reference version by 4, which is the number of CPU cores of the measurement system to enable a more fair comparison. Nevertheless, RT-VQM still outperforms the reference version by a factor between 6 and 8 depending on the resolution.

Moreover, the number of completely measurable video quality layers is evaluated. Notably, the number of processed pixels when measuring a whole SVC stream only depends on the resolution and framerate of the highest video layer and the number of encoded video layers. The reason for this is that all layers are scaled up to the highest layer with respect to resolution and framerate by using the hardware based interpolation method introduced in Section 4 during the Sampling (S) step.

Results for the number of fully comparable video quality layers depending on the highest encoded resolution at 30fps are presented in Table 4. In single precision mode, RT-VQM enables the complete measurement of between 30 (WSIF) and 9 (HD720) video quality layers, i.e., the SVC cube shown in Figure 1 can be assessed entirely in real-time.

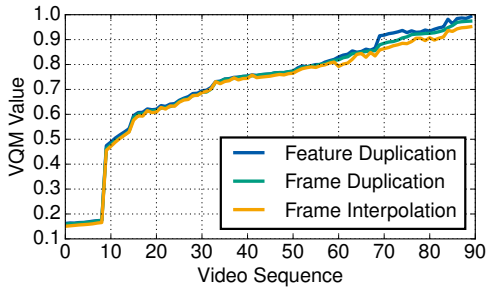
Table 4: Number of fully comparable video quality layers depending on the highest resolution at 30fps in single and double precision mode in real-time.

	WSIF	**WVGA**	**HD720**
Single Precision	30	20	9
Double Precision	24	16	7

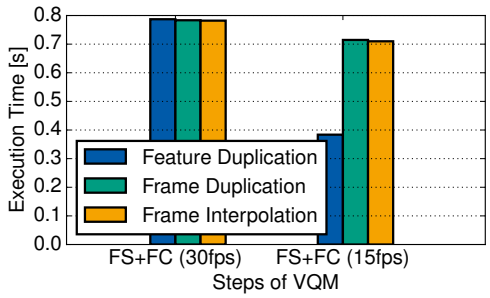
Finally, the precision/performance trade-off for the three modes regarding temporal scaling introduced in Section 4/Figure 9 are evaluated. A measurement of the complete LIVE mobile video database [13] using 30fps for the reference video and 15fps for the downscaled video shows that all three modes do not deviate largely in their resulting VQM score (Figure 9a). In fact, there are no noteworthy differences for scores below 0.8, while above 0.8 the deviation stays below 5%. Nevertheless, the Feature Duplication mode shows considerable execution time advantages compared to the Frame Interpolation and Frame Duplication mode (see Figure 9b). The reason for this is that the amount of features to be processed is much lower than the amount of frames and pixels that have to be processed for the other two modes.

5.3 Influence of Quality Dimensions on VQM

The original VQM algorithm was developed and validated in user studies with respect to quality loss introduced by block artifacts. However, the scalability modes of SVC in-



(a) Comparison of the three modes for handling temporal scaling w.r.t. precision on the LIVE Mobile [13] database. The x-axis denotes the video id in the database. All three modes provide a comparable precision.



(b) Comparison of the three modes for handling temporal scaling w.r.t. execution time E. The Frame Duplication mode provides a considerably better performance than the other modes. The standard deviation is smaller than 1% for all results.

Figure 9: Comparison of the three modes for handling temporal scaling.

troduce block artifacts when scaling along the quantization dimension only, while scaling along the spatial dimension (resolution) for the same display size introduces a loss of contrast (blur) and scaling along the temporal dimension may introduce the loss of motion continuity or the occurrence of flickering effects. It is not entirely clear whether VQM is valid with respect to these dimensions of scalability. However, evaluating the validity of VQM with respect to temporal and spatial scalability is difficult due to a lack of subjective test sets for all three types of impairments.

In order to provide an estimate of the validity of QoE measurements when scaling quality, the quality loss as measured by VQM is correlated with Spatial Information (SI)[2] and Temporal Information (TI) measurements of all videos in the LIVE Mobile [13] database. SI and TI are widely used and well defined measures [5]. Intuitively, the SI measure is a measure for the amount of spatial detail information contained in a video sequence. Increasing spatial complexity results in a higher SI measure. Similarly, the TI measure indicates the amount of temporal change in a video sequence and a higher TI value indicates a more temporally complex scene. For formal definitions, see [5].

Using these measures, the following hypotheses are constructed: (a) An equal reduction of the quantization dimension results in a higher reduction of the VQM value for sequences with a higher SI measure. (b) An equal reduction in frame rate results in a higher reduction of the VQM value for sequences with a higher TI measure. (c) An equal reduction of resolution results in a higher reduction of the VQM value for sequences with a higher SI value.

[2]Not to be confused with the feature set defined in Section 4.

Table 5: Correlation Coefficients for VQM measurements and Spatial/Temporal Information

	SI		TI	
Reduction	**CC**	**SROCC**	**CC**	**SROCC**
Resolution	0.36	0.285	0.382	0.345
Bitrate	**0.686**	**0.648**	0.482	0.43
Framerate	0.176	0.261	**0.785**	**0.648**

For that purpose, we generate four videos for each sequence contained in the LIVE Mobile database [13]. One video is encoded with maximum quality settings, i.e., a target bitrate of $1500^{\text{kBit}/\text{s}}$, HD720 resolution and 30fps. Additionally, three quality degraded versions are generated by reducing one of the following parameters: the target bitrate is set to $250^{\text{kBit}/\text{s}}$ to reduce the quantization dimension, the framerate is set to 15fps to reduce the temporal dimension by dropping every second frame, and the resolution is reduced to WSIF to reduce the spatial dimension. Each of the four videos is compared to the original version from the LIVE Mobile database [13] using RT-VQM. By calculating the pairwise difference between the VQM value of the high quality version and the quality degraded version, the VQM loss along each dimension is quantified.

Notably, a correlation larger than 0.6 could only be found for hypothesis (a) and (b). The correlation value for hypothesis (c) is lower than 0.36 for Pearson Correlation Coefficient (PCC) and Spearman Rank Order Correlation Coefficient (SROCC) (see Table 5) and the measured VQM loss is lower than 0.08 when reducing resolution, i.e., comparably small. This indicates that the VQM model may not react as desired to spatial scaling. Nevertheless, spatial scaling induces mostly blurring and loss of edge information, which are both covered by VQM feature sets. Consequently, a user study should be conducted to quantify, whether the features should be re-weighted before applying the model in practice for quality adaptation along the resolution dimension.

6. CONCLUSIONS

This work proposed an approach to calculate the perceived video quality using objective metrics in real-time and with a high correlation to subjective studies. Therefore, the precise FR-video quality standard VQM was decomposed into its single sub-algorithms in order to identify steps suitable for a massive parallelization using GPUs. Based on this analysis RT-VQM, a highly parallel, GPU supported version of VQM was developed. Therefore, an efficient feature extraction, hardware-based interpolation and a high-performance parallel feature pooling method were introduced. RT-VQM achieves a speedup of up to 30 times in comparison to the VQM reference implementation with only negligible deviations in its precision. Our approach was evaluated with respect to precision, the performance increase, the number of video representations to be compared in real-time and its suitability to measure different modes of video adaptation. The findings show that a precise comparison of up to 9 representations up to HD720 resolution at a framerate of 30fps can be achieved. Moreover, VQM is suitable for quality as-

sessment in the temporal and quality dimension of a video.

RT-VQM enables a number of novel use cases, i.e., the adaptation of live streams and video conferencing streams according to user's perception as well as a more efficient offline processing of VoD content to assess the QoE loss between different video representations upfront or during transmission. Especially the VoD use case proposes interesting opportunities for caching QoE values calculated on-the-fly during the first transmission. Thus, perception optimized quality scaling can be offered for the most important, possibly viral fraction of videos on a short time scale, while assessing long-tail content offline when free processing resources are available, e.g., at night. As a result, video streaming services and content distribution networks can provide a better QoE per invested transmission bandwidth or minimize bandwidth per user in a way that minimizes harms to user perception.

7. ACKNOWLEDGMENT

This work has been supported in parts by the European Union (FP7/#317846, SmartenIT and FP7/#318398, eCousin) and the German Research Foundation (DFG) within the Collaborative Research Center (CRC) 1053 MAKI.

8. REFERENCES

[1] IEEE Standard for Binary Floating-Point Arithmetic (ANSI/IEEE Std. 754-1985). Tech. rep., 1985.

[2] Digital Transport of One-Way Video Signals - Parameters for Objective Performance Assessment (ANSI Std. T1.801-2003). Tech. rep., ANSI, 2003.

[3] Objective Perceptual Video Quality Measurement Techniques for Standard Definition Digital Broadcast Television in the Presence of a Full Reference (ITU-R Recommendation BT.1683). Tech. rep., ITU, 2004.

[4] Objective Perceptual Multimedia Video Quality Measurement in the Presence of a Full Reference (ITU-R Reccommendation J.247). Tech. rep., ITU, 2008.

[5] Subjective Video Quality Assessment Methods for Multimedia Applications (ITU-T Recommendation P.910). Tech. rep., ITU, 2008.

[6] Cuda C Best Practices Guide. Tech. rep., NVIDIA Corporation, 2014.

[7] Global Internet Phenomena Report. Tech. rep., Sandvine Inc., 2014.

[8] Visual Networking Index. Tech. rep., Cisco Inc., 2014.

[9] Chikkerur, S., Sundaram, V., Reisslein, M., and Karam, L. J. Objective Video Quality Assessment Methods: A Classification, Review, and Performance Comparison. *IEEE Trans. on Broadcasting 57*, 2 (2011), 165–182.

[10] Helle, P., Lakshman, H., Siekmann, M., Stegemann, J., Hinz, T., Schwarz, H., Marpe, D., and Wiegand, T. A Scalable Video Coding Extension of HEVC. In *The Data Compression Conference* (2013).

[11] Krishnappa, D. K., Zink, M., and Sitaraman, R. K. Optimizing the Video Transcoding Workflow in Content Delivery Networks. In *ACM Multimedia Systems Conference* (2015).

[12] Larson, E. C., and Chandler, D. M. Most Apparent Distortion: Full-Reference Image Quality Assessment and the Role of Strategy. *Int. Society for Optics and Photonics' Journal of Electronic Imaging 19*, 1 (2010), 1–21.

[13] Moorthy, A. K., Choi, L. K., and Bovik, A. C. Video Quality Assessment on Mobile Devices: Subjective, Behavioral and Objective studies. *IEEE Journal on Selected Topics in Signal Processing 6*, 6 (2012), 652–671.

[14] Movshon, J. A., Thompson, I. D., and Tolhurst, D. J. Spatial Summation in the Receptive Fields of Simple Cells in the Cat's Striate Cortex. *The Journal of Physiology 283*, 1 (1978), 53–77.

[15] Phan, T., Sohoni, S., and Chandler, D. M. Performance-Analysis-Based Acceleration of Image Quality Assessment. In *IEEE Southwest Symposium on Image Analysis and Interpretation* (2012).

[16] Pinson, M. H., and Wolf, S. A New Standardized Method for Objectively Measuring Video Quality. *IEEE Trans. on Broadcasting 50*, 3 (2004), 312–322.

[17] Schwarz, H., Marpe, D., and Wiegand, T. Overview of the Scalable Video Coding Extension of the H. 264/AVC Standard. *IEEE Trans. on Circuits and Systems for Video Technology 17*, 9 (2007), 1103–1120.

[18] Seshadrinathan, K., and Bovik, A. C. Motion Tuned Spatio-Temporal Quality Assessment of Natural Videos. *IEEE Trans. on Image Processing 19*, 2 (2010), 335–350.

[19] Stockhammer, T. Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles. In *ACM Conference on Multimedia Systems* (2011).

[20] Vu, P. V., Vu, C. T., and Chandler, D. M. A Spatiotemporal Most-Apparent-Distortion Model for Video Quality Assessment. In *IEEE Int. Conference on Image Processing* (2011).

[21] Wang, X., and Shi, B. E. GPU Implemention of Fast Gabor Filters. In *IEEE Int. Symposium on Circuits and Systems* (2010).

[22] Wang, Y., Jiang, T., Ma, S., and Gao, W. Spatio-temporal SSIM Index for Video Quality Assessment. In *IEEE Visual Communications and Image Processing* (2012).

[23] Wang, Z., Bovik, A. C., and Lu, L. Why is Image Quality Assessment so Difficult? In *Acoustics, Speech, and Signal.* (2002).

[24] Wang, Z., Bovik, A. C., and Sheikh, H. R. Image Quality Assessment: from Error Visibility to Structural Similarity. *IEEE Trans. on Image Processing 13*, 4 (2004), 600–612.

[25] Wang, Z., Simoncelli, E. P., and Bovik, A. C. Multiscale Structural Similarity for Image Quality Assessment. In *Int. Conference on Signals, Systems and Computers* (2003).

[26] Wolf, S., and Pinson, M. Video Quality Measurement Techniques. Tech. rep., 2002.

[27] Zinner, T., Hohlfeld, O., Abboud, O., and Hossfeld, T. Impact of Frame Rate and Resolution on Objective QoE Metrics. *IEEE Int. Workshop on Quality of Multimedia Experience* (2010).