SCALABLE VIDEO ALGORITHMS FOR RESOURCE CONSTRAINED PLATFORMS

Christian Hentschel

Media Technology, Brandenburg University of Technology Cottbus
Konrad-Wachsmann-Allee 1, 03046 Cottbus, Germany
phone: +49 355 692128, fax: +49 355 692150, email: christian.hentschel@tu-cottbus.de
web: www.tu-cottbus.de/mt/

ABSTRACT

Video signal processing is shifting from dedicated hardware to software implementation due to its flexibility. Digital signal processors (DSPs) for media processing are limited in its resources to enable cost efficient implementations for consumer devices. One way to achieve cost-efficient implementations is to use resource-quality scalable video algorithms (SVAs). This implies that dynamic resource adaptations result in dynamic quality changes which might affect the overall image quality. Starting from properties of SVAs, typical issues on quality including proposals for high-quality image processing will be presented.

1. INTRODUCTION

Algorithms for media processing are usually designed for a specific quality, and for many years implemented on dedicated hardware for their specific environment. For instance, in traditional television receivers various, specific ICs are combined to perform e.g. color decoding for NTSC or PAL systems, noise reduction, or frame rate up-conversion.

It is the trend of technology to develop more and more programmable platforms that allow media applications in software. Expected advantages are reduced time-to-market, re-use of hardware and software modules, portability, and flexibility. These are the issues that gain interest with respect to dedicated hardware.

The always limited computation capabilities are a restriction. This becomes especially a problem for the lower-cost, mass market processors with lower performance. On the software module side, current algorithms are designed for highest quality on given resources. They are not scalable and have a fixed functionality. It is simple to predict that the number of algorithms running in parallel is platform dependent and very limited.

Some internet applications use a kind of scalable algorithms to control the data rate. This is done by subsampling video data, by either deleting entire frames, lines, or pixels. Deleting information together with change of resolution is not acceptable in many application areas such as video processing in consumer television systems.

An alternative is to use resource scalable video algorithms (SVAs) which could solve a number of problems with respect to real-time processing on programmable platforms [1]. Figure 1 shows a range of a programmable product family versus algorithm requirements. Programmable platforms with different resources (figure 1a) will exist in parallel to suit different markets. Current media processing algorithms are designed for highest quality at given resources. In figure 1b, the height of the algorithms illustrates the resources needed for operation. The resource usage and output quality are usually not scalable, meaning that the number of algorithms allowed to run in parallel is platform dependent and very limited. A way of getting beyond these limitations is to design SVAs (figure 1c). These may have a kernel, which is not scalable (dark areas), and a part which is scalable to increase quality (light areas).

New quality issues occur since these scalable algorithms combined with QoS resource management may result in fluctuating quality with a low acceptance rate by the consumer. QoS in networks is already applied, and the differences to terminal QoS for SVAs are subject of the next section. Quality issues combined with resource usage are topics of the other sections.

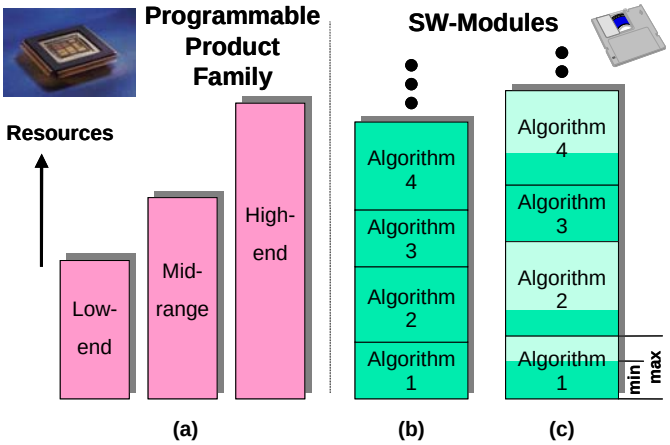

Figure 1: Programmable product family and fixed or scalable software algorithms.

2. TERMINAL QOS VERSUS NETWORK QOS

High-quality video processing in consumer terminals (CTs) has a number of distinctive characteristics when compared to mainstream multimedia processing in, for example, a (networked) workstation environment [2]. Consumer terminals need to connect to various input sources, and are increasingly being integrated in wired and wireless network environments. The transmission of various data streams including graphics, audio and video over networks started in the workstation and PC domains. From a single user point-of-view, data transmission requests are often seen as point-to-point transmissions. Network requests from other users are independent and these additional activities are recognized by the single user only because of long delays or even network access denials. The most limiting resource is network bandwidth, which has to be shared by all current users. To solve these transmission problems, QoS has been introduced to optimise the service between different users. Data streams may get priorities and a specific portion of the network bandwidth. Typical QoS parameters for streaming video over networks are image resolution, image size (window), frame rate, color depth, bit rate and compression quality in order to lower the transmission bandwidth. In summary, network QoS trades transmission bandwidth resources to optimise the overall quality.

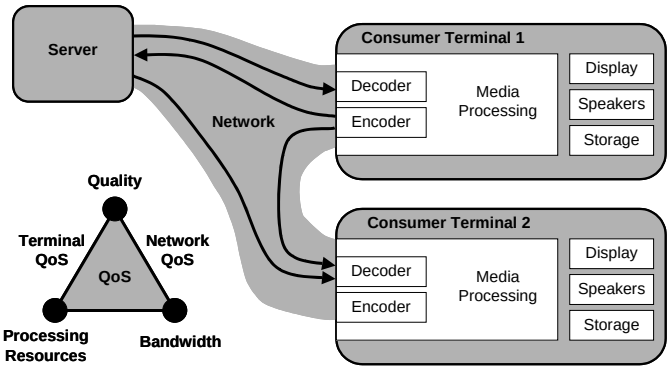

Figure 2: Differences between terminal QoS and network QoS.

Figure 2 shows an example of two future consumer terminals and a server in a network environment [3]. These terminals provide decoders and encoders, which are parts of the media processing and network interfaces. Output devices may be displays, speakers and storage devices. The different applications (e.g. view a movie, access the internet, play games, etc.) require different media processing algorithms, which also depend on the input data (resolution, frame rate, quality, etc.).

Typically, the processing resources are the limiting factor in consumer terminals, and not the bandwidth resources. Therefore, QoS in consumer terminals is different from that in networks. Multimedia con-

sumer terminals usually have a fixed resolution. The image size is determined by the display or chosen by the end-user, but not by the system. Consumer terminals have real-time requirements and do not allow frame rate fluctuations or audio interruptions.

As depicted in figure 2, terminal QoS trades processing resources over quality. The triangle also connects processing resources with bandwidth. An example for the validity of this triangle is an MPEG transmission. With the given transmission bandwidth, the data quality can also be influenced by the encoder processing resources. With more processing resources, a higher quality can be achieved at the same transmission bandwidth.

CTs such as TV sets and Set-top boxes are currently receivers in a broadcast environment, and therefore do not have the option to negotiate compression quality and bit-rate, although that may change in the future for CTs in an in-home digital network.

3. PROPERTIES OF SCALABLE VIDEO ALORITHMS

A scalable algorithm is an algorithm that:

- allows the adaptation of quality versus performance on a given architecture,
- supports different software and/or hardware platforms for media signal processing, and
- is easily controllable by a control device for several predefined settings.

A set of scalable algorithms in a modular form can perform the different applications needed in a set-top box, TV set, multimedia PC, or, more in general, media processing unit.

3.1 Basic scalable video algorithms

Figure 3 shows a block diagram of a scalable algorithm. The algorithm for media processing consists out of different functions. Some of them are scalable for several quality levels, but there is no need that all of them have to be scalable. The outcome of the scalable algorithm is dependent on the appropriate combination of the quality levels of the functions (Figure 4). These combinations may vary a lot, but only few of them may provide acceptable quality levels for the scalable algorithm.

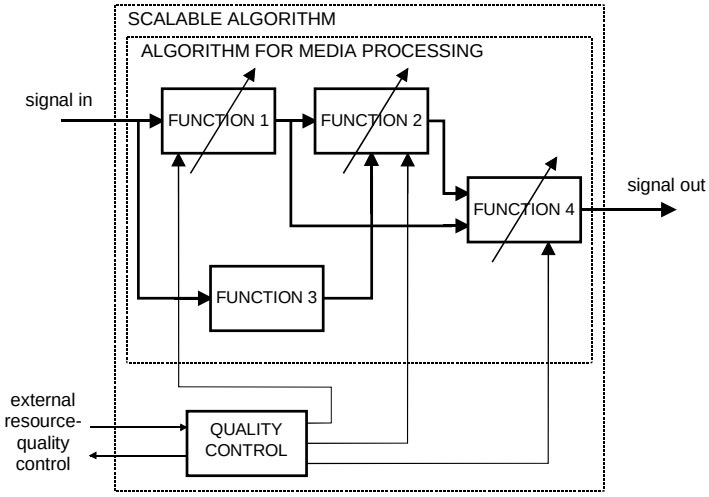

Figure 3: Basic structure of a scalable algorithm.

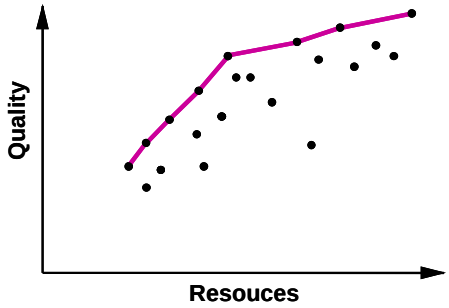

Figure 4: Best choices of quality-resource combinations for functions of the entire scalable algorithm.

The control signal for quality level can be as simple as the chosen quality level or guaranteed budget/resources (Figure 3). The block 'quality control' has specific knowledge about the algorithm for media processing and the best combinations of settings for the several functions. Because of this specific knowledge, it becomes an important part of a scalable algorithm.

3.2 Example of a simple SVA with data-independent resource usage

The block diagram of a scalable down-scaler is shown in figure 5. Picture-in-picture (PiP) applications require only simple down-scaling by natural factors, so the algorithms only includes a low-pass filter, followed by the down-sampling [3]. In this case, the down-scaling is restricted to a factor of four, and just the low-pass filter is scalable. Only output pixels need to be calculated, which reduces processing resources significantly.

The scalable down-scaler with a decimation factor of 4 in both the horizontal and vertical directions requires between 4-14 MIPS (table 1). At the lowest quality level QL0, subsampling alone without any pre-filtering is performed. Quality level QL1 uses an average filter over 4x4 pixels for the luminance, while QL3 uses the same filter for the chrominance, too. Separable 5-tap filters for the luminance (QL2) and for both luminance and chrominance (QL4) complete the scalability range (table 1). The scalability ranges from 29-100 %, corresponding to a resource range of 10 MIPS. Quality in figure 6 was coarsely estimated by a few expert viewers.

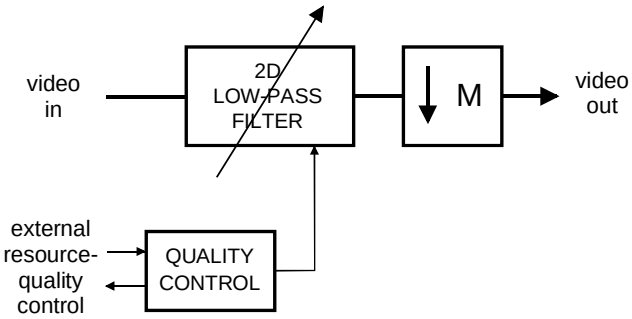

Figure 5: Basic structure of scalable image down-scaler.

Quality Level	Resource Usage [MIPS]	Low-Pass Filter Luminance	Low-Pass Filter Chrominance
0	4	off	off
1	8	average over 4x4 pixel	off
2	9	separable filters over 5x5 pixel	off
3	12	average over 4x4 pixel	average over 4x4 pixel
4	14	separable filters over 5x5 pixel	separable filters over 5x5 pixel

Table 1: Quality levels and functional details of the resource-scalable spatial down-scaler

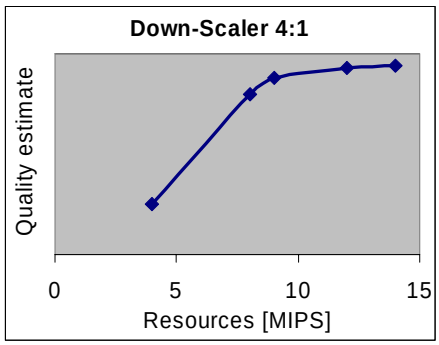

Figure 6: Best choices of quality-resource combinations for functions of the down-scalers for PiP applications.

Scalable algorithm with quality scalability in exchange to compute requirements have typically an essential core to perform its minimal function. The quality can then be increased, depending on the available computation resources. Typical quality-resource behaviour of an SVA is shown in figure 7. At minimal resources the quality can be very low, with a steep increase in quality for increasing resources. An example is a spatial scaler with low resource usage by pixel subsampling (down-scaling) or repetition (up-scaling) with poor output quality. A bilinear interpolation with few additional resources increases the quality significantly. Further quality improvements by using high-order polyphase filters become smaller compared to the resources required. As a result, a wide range of resource scalability is possible within a small range of quality. The area of high quality changes at small resource changes should be avoided for scalable media processing.

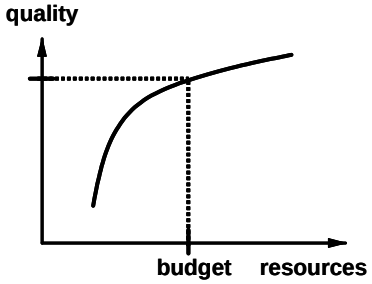

Figure 7: Typical quality-resource behaviour of scalable algorithms.

The resource-quality behaviour in figure 7 is simplified and cannot reflect all real properties. Resources are multidimensional and can include CPU cycles, memory, bus bandwidth, coprocessors, etc. Quality can be measured in a multidimensional space as well. Properties such as sharpness, color reproduction, noise, quantization, algorithm specific compression artefacts can be expressed in quality, but also more specific properties such as resolution in x- and y-direction, temporal resolution, judder, etc. A research topic is how these parameters influence each other, especially in a dynamic environment.

3.3 SVAs with data dependent resource requirements and quality

Programmable components are most suitable for irregular processing. Compression algorithms and non-linear processing algorithms such as motion estimation have such irregular processing [4], while video processing algorithms such as scaling, image enhancement etc. are mostly executed by pipelined regular pixel processing. Thus video processing on programmable components require different kinds of algorithms to ensure efficient and effective processing at available resources.

A different processing approach is illustrated in figure 8. The advantage of irregular processing is the option to do input data dependent image analysis and choose for a strategy how to process the data. An example is priority processing of sharpness enhancement in images. All relevant edges must be detected which could appear sharper by adding detail information. In flat, unstructured regions the addition of detail information would increase the noise level which lowers subjective image quality. With priority

processing, first the most relevant edges should be enhanced, followed by lower priority regions. Flat regions should not be processed, or, in case of still available resources, could be processed by noise reduction algorithms.

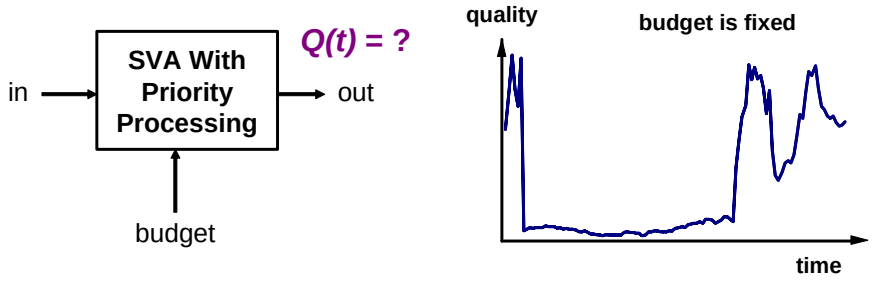

Figure 8: Fixed processing budget/resources and data dependent output quality.

An advantage of priority processing is to be able to interrupt image processing in case of low resources while still get the maximum on image quality. Depending on the image content (heavily structured or more flat areas) resources for a fixed output quality vary and vice versa (Figure 8). Processing resources have no fixed relationship to output quality and cannot be used for quality estimation. Therefore, output quality measurement becomes an important subject for resource allocation and overall system and application optimization.

Figure 9 shows an SVA with internal quality measurement to indicate its data and budget (resource) dependent performance. Typically, media processing functions can be used to estimate the output quality. In case of sharpness enhancement, the final priority level processed gives an indication of the achieved output quality. For motion estimation, the final, average accuracy and reliability can indicate the output quality.

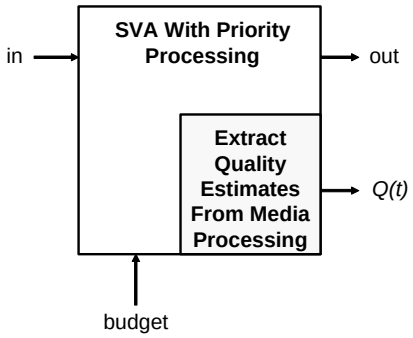

Figure 9: Quality measure within a scalable video algorithm.

3.4 SVA with content dependent resource fluctuations and load balancing

The general schematic diagram of a progress-based resource regulator is shown in Figure 10 [5]. The basic algorithm for media processing contains two new functions for the measurement of progress *P* and used resources *Rr*. These measurements are derived from internal media specific processing data and are therefore independent from external system data. For example, in the 3D-RS motion estimator, *P* is the number of block lines that have been processed and *Rr* is the number of vector candidates that have been used (which is a good measure for the actual number of CPU cycles used [4]).

The external input indicates the available budget per assigned period, typically a frame. For the motion estimator example, this is the total number of vector candidates that is, on average, needed for processing all block lines within a frame.

Together with the measured progress *P*, the expected resource usage can be calculated at incremental periods of a frame. This expected resource usage can then be compared with the measured resource usage *Rr* to calculate the deviation from target *Rd*. The calculated target may be smoothed by a low pass

filter and additionally corrected by a non-linear function before adjusting the resource/quality setting of the scalable media processing algorithm.

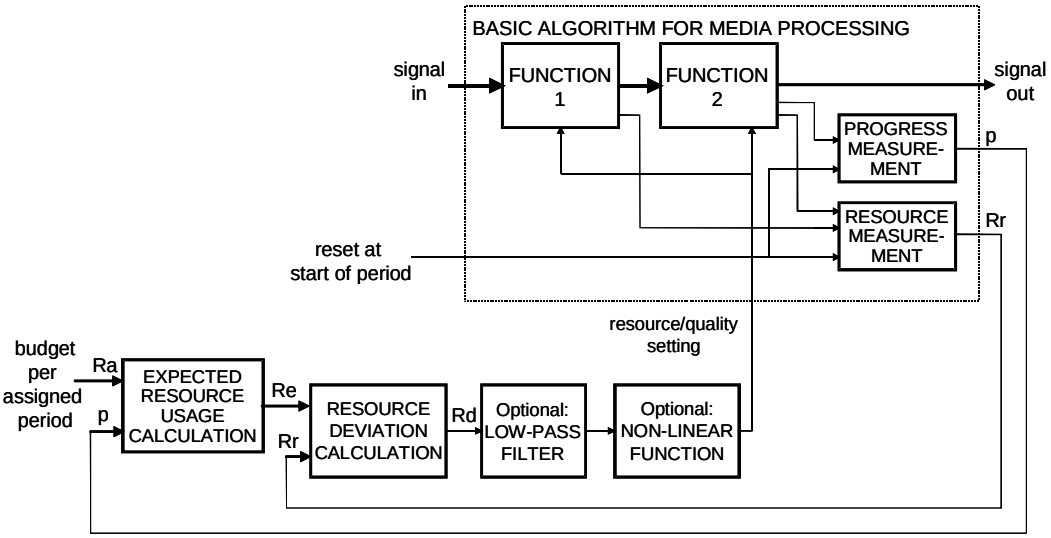

Figure 10: Schematic diagram of the progress-based resource regulator.

This regulation scheme ensures that the regulation properties are independent of the amount of data already processed (actual progress, e.g. screen position). The regulation works on differences between expected and real resource usage during the assigned period and regulates close to a specific resource budget per frame, independent of the input data properties. Despite the very good regulation properties, also a quality gain has been recognized by observers.

4. QUALITY ISSUES IN AN SVA CHAIN

A modular system approach with simple QoS resource management is essential for flexible and manageable multimedia consumer terminals. Consumer terminals include mobile and stationary devices with stand-alone capabilities or within a network environment. Instead of measuring the quality at several positions of a processing chain, SVAs with integrated quality measurement could significantly simplify the system design.

In addition, properties of the input signal can help to optimize the functional structure as well as the quality of the overall system. Input signals come from different sources such as analog NTSC or PAL sources, Y/C, digital sources with different compression algorithms and compression quality (MPEG, DV, H.264…) and so on. These input signals may have specific artifacts such as noise (SNR), blocking, ringing, etc., could vary in resolution (HD, SD, CIF,…) and sampling formats (4:4:4, 4:2:2, 4:2:0, 4:1:1…). Since these input signal properties are important for the entire chain processing, these parameters should be collected in the block ‘video analysis’ close to the video decoder as depicted in figure 11. Analysis information can then be used for optimizing quality in the resource-scalable processing chain.

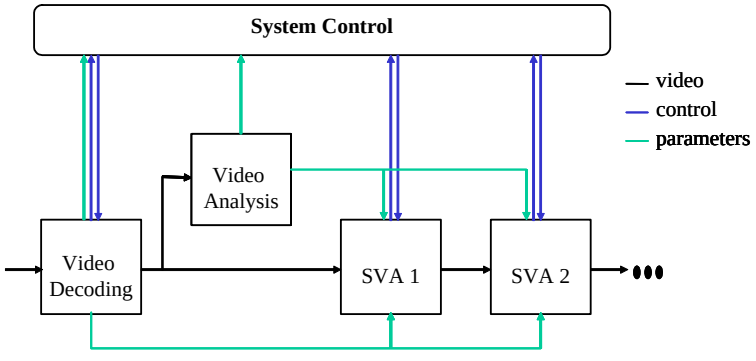

Figure 11: Input signal quality analysis for quality control of the processing chain.

5. CONCLUSIONS

In the past image quality has often been optimized on single processing algorithms which later were combined for the entire application. The result of the final quality often depended on the experience and knowledge of the engineers. The signal source, encoding, and transmission already affects image quality, which should be preserved as much as possible. In the receiver, incoming image quality as well as processing steps including the order of certain algorithms have to be taken into account for optimal image quality. A new approach is the development of scalable video algorithms suitable for programmable components. Simple mapping of algorithms designed for ASIC implementation cannot provide cost-effective and efficient usage of programmable components. Starting from basic SVAs, examples of SVAs for data depending processing resources including a novel approach with load balancing were described. Investigations showed the effectiveness of these methods. Since image quality in scalable applications will dynamically fluctuate, new, real-time methods are required to optimize the overall output quality. Video analysis at an early stage in the processing chain can provide information about the signal source parameters which can be used for optimizing resource distribution over the processing components.

REFERENCES

[1] C. Hentschel, R.J. Bril, M. Gabrani, L. Steffens, K. van Zon, S. van Loo, *Scalable Video Algorithms and Dynamic Resource Management for Consumer Terminals*, Int. Conf. on Media Futures (ICMF 2001), Proceedings, Florence (Italy), May 2001, pp. 193-196.

[2] K. Nahrstedt, H. Chu, S. Narayan, *QoS-aware Resource Management for Distributed Multimedia Applications*, Journal on High-Speed Networking, Special Issue on Multimedia Networking, IOS Press, Vol. 8, No. 3-4, pp. 227-255, 1998.

[3] C. Hentschel, R.J. Bril, Y. Chen, R. Braspenning, Tse-Hua Lan, *Video Quality-of-Service for Consumer Terminals - A Novel System for Programmable Components.* IEEE Transactions on Consumer Electronics Vol. CE-49 (2003), No. 4 (November), pp. 1367-1377.

[4] R. Braspenning, G. de Haan, C. Hentschel, *Complexity Scalable Motion Estimation*, International Conference on Visual Communications and Image Processing (VCIP), Proceedings, San Jose (USA), January 2002, pp. 442-453.

[5] C. Hentschel, R. Wubben, *Novel Resource Regulator for Media-Processing Algorithms on Programmable Components*, International Conference on Consumer Electronics, Digest of Technical Papers, Las Vegas (USA), Jan. 10.-12., 2005, pp. 443-444.