A 3D-DCT Real-Time Video Compression System for Low Complexity Single-Chip VLSI Implementation

Andreas Burg †, Roni Keller ‡, Juergen Wassner †, Norbert Felber †, Wolfgang Fichtner †

Integrated Systems Laboratory, Swiss Federal Institute of Technology (ETH) Zurich
† apburg, wassner, felber, fw@iis.ee.ethz.ch
‡ roni.keller@gmx.ch

Abstract:

This paper presents the first VLSI implementation of a real-time color video compression/ decompression system, based on the three-dimensional discrete cosine transform (3D-DCT). Compared to motion-estimation/compensation based algorithms, the 3D-DCT approach has three major advantages:

- No motion estimation is required, greatly reducing the number of en/decoding operations per pixel.
- En- and Decoder are symmetric with almost identical structure and complexity, which facilitates their joint implementation.
- The complexity of the implementation is independent of the compression ratio.

These factors are key issues for the realization of mobile video compression systems.
We describe the system architecture and implementation. Tradeoffs that facilitate the VLSI implementation are emphasized and performance results are presented.

Keywords: 3D-DCT, video compression, VLSI, motion prediction

Introduction:

The increasing demand for portable digital video applications as cellular video phones or fully digital video cameras is pushing the need for highly integrated real-time compression and decompression solutions.
For these types of applications, efficient low-cost/complexity implementation and low power consumption are the most critical issues.
Most of today's established compression standards like MPEG-2/4 and H263(+) rely on the so called motion-estimation/compensation approach to exploit interframe correlation. This is a highly complex process, which requires a large number of operations per pixel and is therefore less appropriate for the implementation as real-time compression in a portable recording- or communication device. Moreover, even though very efficient for scenes with a translatoric character, the algorithm loses much of its efficiency in scenes of morphological character or with extremely fast moving objects.

An alternative approach to prediction-based exploitation of interframe correlation is to use a transform-based approach for the encoding of subsequent frames. The goal hereby is to find an appropriate transform that has the desired energy-compaction property and therefore allows an efficient subsequent entropy encoding. For the compression of still images the Wavelet transform and the 2D-DCT transform have been found to have this property. Particularly the latter is widely used in most modern image/video compression algorithms in the spatial domain. The 2D-DCT has the potential of easy extension into the third dimension, i.e. 3D-DCT. The proposed solution is based on this approach. It includes the time as third dimension into the transformation and energy compaction process.

In section 1 we review the basics of the 3D-DCT compression algorithm. A more detailed introduction can be found in [1,10]. Section 2 presents the complete system and the extensions chosen to improve the compression ratio. It also emphasizes the tradeoffs required for efficient VLSI implementation. This is the main contribution of this work. Section 3 gives an overview of the ASICs architecture.
Results, obtained with a bit-true software implementation of the chip, which is currently being integrated, are given in section 4.
The last section concludes this paper with a short summary.

Section 1: The 3D-DCT

Figure 1 shows an overview block diagram of the compression and decompression process.

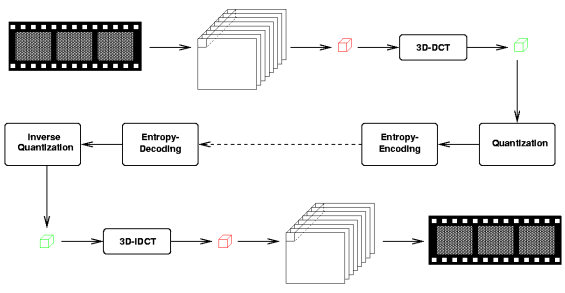

Figure 1 3D-DCT Flowgraph

First a sequence of 4 or 8 subsequent frames is collected and partitioned into three-dimensional video-cubes. The size of the cubes in our implementation is chosen to be 8*8 pixels in the spatial domain. On the time axis a depth of either 4 or 8 frames is being used. While deeper cubes lead to higher compression ratios, they also increase the latency of the compression process and the memory requirements.

Secondly, the cubes are transformed into the frequency domain, using a three-dimensional cosine transform. The energy compaction property of the cosine transform concentrates most of the information in a few low-frequency components. An example for the probability distribution after the transformation is shown in Figure 2.

Subsequent quantization selectively removes information that is less relevant to the observer. This mostly affects the higher-frequency components. The quantization process determines the tradeoff between quality of the result and compression ratio. As quantization has an impact on spatial as well as on temporal components of the sequence, an additional degree of freedom which allows trading quality for compression ratio without affecting the computational complexity is attained. A detailed analysis of optimized quantization matrices based on rate-distortion theory and experiments can be found in [4,5] for the 2D case, while [3] describes a statistical approach for the 3D case. Other proposals have been made for 3D-DCT quantization schemes [6,7]. However they mostly aim at the compression of 3D images as found in medical imaging applications and apply different criteria for the quality measurement of the result.

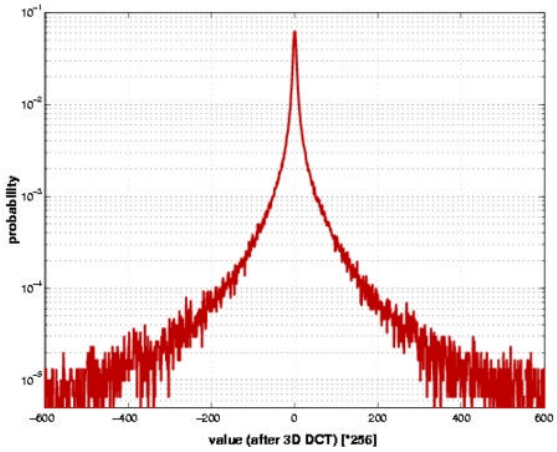

Figure 2 Probability distribution after 3D-DCT

Finally, compression is performed in our approach with a zero run-length encoder and subsequent variable-length encoding.

For the reconstruction process the above steps are just inverted and carried out in reverse order.

The similarity of the encoding and decoding process is one of the interesting properties of the 3D-DCT compression since it greatly facilitates the implementation of a joint en-/decoder.

Section 2: System description

This section gives a more detailed description of the parts of the system outlined above. It also describes some extensions of the basic algorithm that lead to improved compression results. Tradeoffs are presented that greatly simplify an efficient implementation while only slightly degrading the overall performance.

1. Color-space conversion:

As the human visual system is less sensitive to color (chrominance) than to contrast (luminance), the original RGB image is initially converted into the YCbCr color space, separating the luminance component (Y) of the signal from the two chrominance components (Cb and Cr). This separation allows to apply a more aggressive quantization to the less sensitive color components of the image sequence.

2. Differential Prediction:

Differential encoding is the simplest possible prediction scheme. It proves especially efficient for scenes with highly detailed static backgrounds. Since the 3D-DCT intrinsically decorrelates successive frames within individual video cubes, differential encoding within cubes is inappropriate. However, since the transform does not operate across cube boundaries, differential prediction may be used to remove some of the redundant information in subsequent cubes. Hereto we employ the last frame of the known cube as a predictor for each frame in the following cube. As the same predictor is used for every frame, this process only affects the temporal DC-component in the frequency domain. Especially in relatively static scenes these components carry the highest amount of information. The prediction process removes redundant information from the DC-plane and therefore greatly improves the achievable compression ratio.

A major problem of combining differential encoding with a lossy compression algorithm is the fact that the original predictor is not perfectly known in the receiver. A general solution to this problem is to reconstruct the predictor in the transmitter after quantization, just as it is seen by the receiver, and to use this reconstruction instead of the original frame. This is depicted in Figure 3. The drawback of this approach is the need to concurrently run encoding *and* decoding in the transmitter, which nearly doubles complexity.

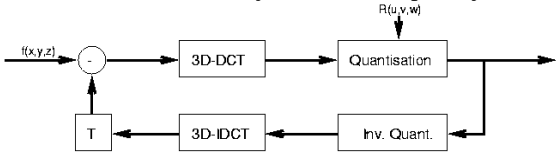

Figure 3 Feed-back prediction loop

To avoid this overhead our VLSI implementation uses the feed-forward loop shown in Figure 4.

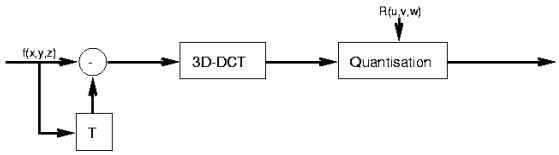

Figure 4 Feed-forward prediction loop

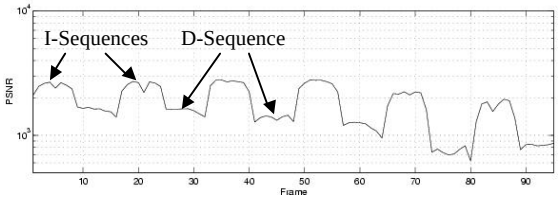

Figure 5 PSNR drop in a feed-forward encoded sequence (Miss-America, 1:130)

This simplification introduces a 3dB quality loss in all frames of differentially encoded sequences (D-sequences). To prevent propagation of the introduced error, only frames from sequences which are not differentially encoded (I-sequences) are used as predictors. This yields a step-like run of the peak signal-to-noise ratio (PSNR) curve as shown in Figure 5. Here, the additional PSNR drop after frame 72 is caused by increased motion in the video sequence.
A specific predictor can be used for many subsequent D-sequences without introducing an additional error, although the prediction becomes less efficient as the similarity between the target frames and the original predictor vanishes with time.

3. DC-Prediction:
After the transformation into the frequency domain, the information is concentrated in the low-frequency coefficients. Looking across cube boundaries in the spatial domain, it becomes obvious that the DC-components of adjacent cubes can be highly correlated, especially in scenes with large monotonous areas. This correlation is again not exploited by the 3D-DCT due to its limited scope. A good prediction of the DC-component of some cube could be obtained from a combination of the DC-components of all its neighboring cubes. However, with respect to the implementation, only the previous cube to the left is used as a predictor, with the DC-component of the first cube in every row being encoded with its absolute value. This corresponds to the feed-back prediction scheme in Figure 3. Thus, error propagation is avoided for the DC-components. In this case the introduced overhead is marginal.

4. Quantization:
The quantization step allows the tradeoff between quality and compression ratio. While in the 2D-case quantization only offers two degrees of freedom to trade spatial blur for compression ratio, it offers three degrees of freedom in the 3D case.
Very little work has been done so far in exploring the sensitivity of the human visual system to quantization induced distortions of the temporal frequency components. In our approach we simply use an exponential degradation along the spatial as well as along the temporal axis as quantization pattern. Another approach to this problem is described in [3] where statistical properties of the original signal are used to derive the quantization pattern.
An additional degree of freedom is provided by independently quantizing the luminance and chrominance channels, which can be treated independently. In general the color information in the Cb and Cr channels can be compressed much more than the contrast information, without having a noticeable effect on the observer.

5. Streaming:
The streaming process converts the 3D data structure of the video cube into a linear data stream. The stream order is essential for the subsequent zero run-length encoder. To optimize its efficiency the stream order has to maximize the average number of subsequent zeros. As the statistical properties of each component within a cube are known, the optimum sequence can be determined by sorting the components with their zero probability in ascending or descending order.
If latency is not an issue, the streaming process can be delayed until all cubes of the sequence are compressed. The optimized stream order can then be applied to the whole sequence. This, again, increases the compression ratio and facilitates the use of component-specific error protection mechanisms for the transmission over lossy channels and the use of variable data rate channels. However, since the latency penalty and the hardware effort to store the whole sequence are very high, in this implementation each cube is streamed out individually.
[9] suggests diagonal planes as good approximation of an optimized stream order. A more complex order, which is tightly coupled to the suggested quantization pattern, has been described in [3]. Our experiments show that using diagonal planes yields good results for many different types of quantizations. This approach also allows a simple VLSI implementation without the need for area consuming lookup tables or complex computations. It is therefore used for the hardware implementation.
An appropriate way to reorder the components of the three color channels has to be found. Due to more aggressive quantization of the two chrominance channels they usually have a higher probability of becoming zero than the corresponding component in the luminance channel. To accommodate this, the Y-channel is

streamed out first, followed by the interleaved Cb and Cr channel, as described below:

$$Y_{DC}, Y_{AC1}, Y_{AC2}, Y_{AC3}, Y_{AC4}, Y_{AC5}, \cdots$$
$$Cb_{DC}, Cr_{DC}, Cb_{AC1}, Cr_{AC1}, Cb_{AC2}, Cr_{AC2}, \cdots$$

6. Run-Length (RL) Encoding:

The first step in the actual compression process is zero RL encoding. It combines runs of successive zeros into 2-tuples of a leading zero and the number of zeros in the run. With the knowledge of the statistics of the non-zero components, the encoding of the RL descriptors can be optimized for the subsequent variable-length encoding. The symmetric statistical distribution of the DCT components in Figure 2 encourages the use of a symmetrical encoding of the RL-descriptor: encoding even runs with a positive number and odd runs with a negative number. Shorter runs will be encoded with shorter symbols, while longer RL-descriptors are used for longer runs.

7. Entropy-Encoding:

A standard Huffman encoder is being used for the variable-length encoding. As the probability distribution of the signal is nearly symmetrical, a fully sign-symmetrical code-book is chosen. Its size is limited to a range of [-255,+255] to maintain a reasonable size. All values that exceed the range of the code-book are encoded with a unique prefix code, followed by their actual value.

Our experiments show that the optimum code-book is relatively independent of the type of scene that is being compressed, while it is clearly subject to the quantization level used. The en/decoder therefore contains three built-in code-book ROMs for low, medium and high compression ratios. An additional sequential decoder with a programmable code-book can be used when the system does not have to run at its full speed. It is mainly intended for evaluation purposes. The limited range of the code-book has no noticeable influence on the compression efficiency.

Section 3: VLSI Implementation

The goals for implementing the 3D-DCT algorithm were threefold:

- Realize the first real-time VLSI implementation of this algorithm (to the best of our knowledge).
- Provide a flexible test-bed for the algorithm by leaving some degree of freedom to tune different parameters and test their influence on the compression results in a real-time environment.
- Demonstrate that the 3D-DCT algorithm can in fact be realized with a very moderate level of complexity.

Figure 6 shows a simplified block diagram of the design.

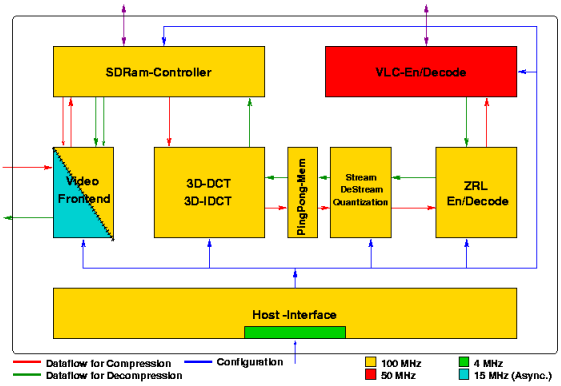

Figure 6 IC-Block Diagram

The front-end is designed to directly interface with different standard PAL/NTSC en/decoders. It runs at the clock rate of the video source, fully asynchronous to the rest of the system. It performs pre-/postprocessing like color-space conversion and differential encoding. A standard SDRAM is being used to store the number of frames required for the 3D-DCT sequence and to buffer the I-frames for differential encoding. The 3D-DCT transformation as well as subsequent quantization and entropy coding is carried out at a 100MHz clock rate. This provides enough speed for standard image sequences with a resolution of 768*576 pixels at a rate of 25 frames per second. The 3D-DCT operation is realized as a sequence of optimized 1D-DCTs, which are again decomposed into so called Butterfly operations. The similarity of the DCT and its inverse enables reusing the complete arithmetic for the decompression mode. This allows a joint en/decoder implementation at virtually no extra cost. The implementation works with cube depths of 4 or 8 frames, offering the potential to adapt to different latency requirements. For test purposes the 2D-DCT is also supported.

The system is equipped with an on-chip memory that holds the configurable quantization tables of the three color components. For the Huffman encoder it contains three built-in ROM code-books for en/decoding at different quantization levels, and an on-chip RAM for a programmable code-book. For the configuration of the ASIC, a standard SMBus interface has been implemented [11]. A second high-speed synchronous serial interface allows very fast reconfiguration of the compression parameters at runtime.

Figure 7 shows the layout plot of the chip and Table 1 gives an overview of the functional and physical data of the design.

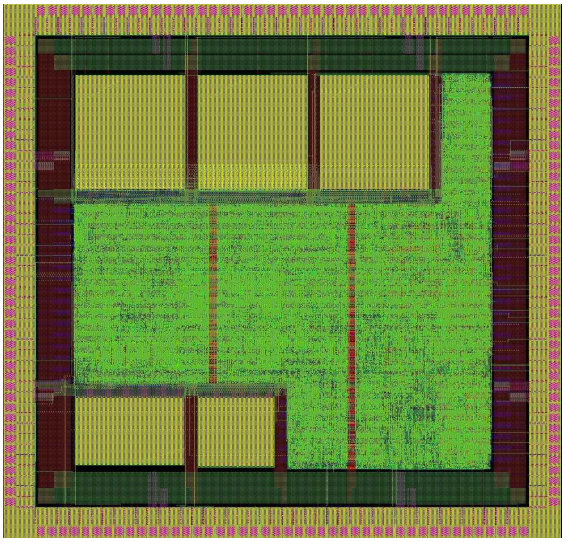

Figure 7 Layout plot

Table 1 3D-DCT ASIC Overview

Functional data:	
Compression & Decompression	
Video Bandwidth up to 15MHz (>PAL/NTSC, 768*576*25Hz)	
Cube-depth: 4 or 8 frames	
Differential Encoding (1-256 D-sequences per I-Sequence)	
DC-Prediction	
Configurable Quantization tables for Y,Cb,Cr	
Configurable VLC-code-book	
Physical data:	
Video Interface:	Programmable 15 MHz, RGB/YcbCr
System Clock:	100 MHz
Transistors:	1,200,000
On-chip memory:	92 kbit
Chip Area:	27.5 mm^2 (pad limited)
Process:	0.35 µm, 5-metal

Section 4: Results

This section gives a brief summary of results obtained from bit-true simulation of the system described above.

Table 2 presents PSNR and compression ratio for the "Miss America" sequence [12] for medium and strong quantization. It also shows the gain in compression ratio achieved by the prediction of DC-components, and the impact of differential encoding on compression ratio and PSNR. Due to its static nature and morphological character of the motion the scene is well suited for the 3D-DCT and can profit from DC-prediction and differential encoding. The compression ratio that we could achieve with an MPEG-2 encoder was 1:270 with a PSNR of 30dB, which is less than the results achieved with the 3D-DCT. Figure 9 shows a comparison with the MPEG result on the left half of the frames and the 3D-DCT result on the right half.

Table 2 Miss-America Sequence (10 fps)

	3D-DCT	+ Diff-Encoding	+ DC-Prediction	+both
Ratio(medium)	1:50	1:61	1:59	1:69
PSNR	36dB	35dB	36dB	35dB
Ratio(strong)	1:117	1:181	1:196	1:277
PSNR	33dB	32dB	33dB	32dB

Table 3 shows the corresponding results for the "Carphone" [12] test sequence. It contains more motion and a more detailed background.

Table 3 Carphone Sequence (10 fps)

	3D-DCT	+ Diff-Encoding	+ DC-Prediction	+both
Ratio(low)	1:8	1:10	1:9	1:11
PSNR	40dB	39dB	40dB	39dB
Ratio(strong)	1:73	1:92	1:82	1:102
PSNR	27dB	26dB	27dB	26dB

Figure 8 plots the PSNR versus compression ratio of both test sequences for different quantization patterns with and without prediction.

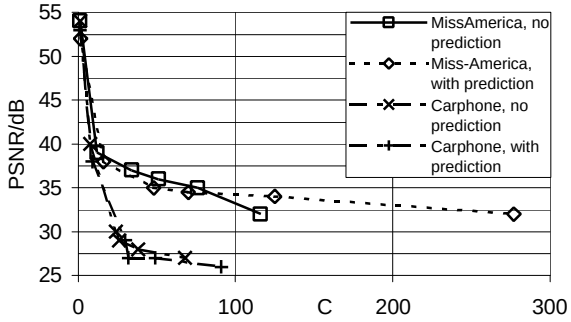

Figure 8 PSNR for "Miss-America" & "Carphone"

Section 5: Conclusions

The straightforward transform-based approach and the simplicity of the underlying algorithms greatly facilitate an ASIC implementation. Compared to motion estimation based encoders, for example the H263 encoder implementation presented in [8], the overall complexity of our final design is very low. Reducing the numerous configuration options of this particular implementation to a necessary minimum would further significantly reduce the implementation cost. When low complexity on the compression side is a key issue the 3D-DCT approach provides an interesting alternative to today's more prevalent algorithms.

In this project only little effort has been put into the optimization of power consumption of the ASIC and into optimization of quantization patterns. The second issue shall be addressed with the presented implementation in a real-time environment.

Future work will have to address QoS aspects and problems associated with the transmission over lossy channels. Nevertheless we believe that the 3D-DCT is an interesting alternative to common compression schemes, and still offers potential for future development and improvements.

Acknowledgments

We would like to thank the Integrated Systems Laboratory of the ETH-Zurich for funding this project and C. Balmer for taking care of the layout finishing and the backend verification of the design.

References

[1] R. Westwater, B. Furth, “Real-time video compression”, Boston Klumer Cop., 1993

[2] M.P. Servais, G. De Jager, “Video Compression using the Three Dimensional Discrete Cosine Transform”, Proc. COMSIG, 1997, pp. 27-32

[3] M. C. Lee, K. W. Chan, D. A. Adjeroh, “Quantization of 3D-DCT coefficients and Scan Order for Video Compression”, Journal of visual communication and image representation, Vol. 8. No. 4, Dec, pp.405-422, 1997

[4] H. Lohscheller, “A subjective adapted image communication system”, IEEE Trans. Comm. COM32-(12), Dec 1984

[5] H.A. Peterson, “An improved detection model for DCT coefficient quantization”, SPIE Proceedings 1913, 1993, pp.191-201

[6] H. Lee, Y. Kim, A. H. Rowberg, E. A. Riskin, “Statistical distribution of 3D-DCT coefficients and their application to interframe compression algorithm for 3-D medical images”, IEEE Trans. on Medical Imaging, Sept. 1993

[7] V. Chameroy, R. Di Paola, “Towards multidimensional medical image coding”, SPIE Vol. 1653, Image Capture, formatting and Display, 1992,pp.261-269

[8] M. Harrand, J. Sanches, “A Single-Chip CIF 30-Hz, H261, H263, and H263+ Video Encoder/Decoder with Embedded Display Controller”, SSC Vol. 3, Nr. 11

[9] B.L. Yeo, B. Liu, “Volume rendering of DCT-based compressed 3D scalar data”, IEEE Trans. on Visualization and Computer Graphics, 1(1), March 1993, pp. 285-296

[10] A. Burg, R. Keller, “A Real-Time Video Compression System Based on the 3D-DCT”, Diploma-Thesis, Integrated Systems Laboratory ETH-Zurich, 1999/2000

[11] “System Management Bus Specification, Rev. 1.1”, Intel Corp., http://www.sbs-forum.org

[12] Test-Sequences: ftp://sotka.cs.tut.fi/cost/Ossi/seq

Figure 9 Example: Miss-America frames 8,12,16,20. MPEG: left half of frames (1:270), 3D-DCT: right half of frames (1:277)