

# DYNAMIC CONFIGURATION: BEYOND VIDEO CODING STANDARDS

Iain Richardson<sup>1</sup>, Maja Bystrom<sup>2</sup>, Sampath Kannangara<sup>1</sup> and Manuel de Frutos Lòpez<sup>3</sup>

<sup>1</sup>The Robert Gordon University, Aberdeen, UK

<sup>2</sup>Boston University, Boston, US

<sup>3</sup>Universidad Carlos III de Madrid, Madrid, Spain

## ABSTRACT

Video coding standards are essential to a wide range of applications, but have certain limitations. New, flexible approaches to video compression have the potential to speed up adoption of new coding techniques, to improve compression performance through adaptive coding and to support multiple coding standards on a single platform. In this paper we describe the MPEG Reconfigurable Video Coding initiative, which enables flexible configuration of coding tools drawn from a standard Video Tool Library. Further, we present an alternative framework, dynamically configurable video compression, in which the coding tools themselves may be created, configured and re-configured adaptively. We present an initial prototype of this framework and discuss performance results and research challenges.

## I. INTRODUCTION

Existing video communication systems use fixed, pre-defined and typically standardized methods of coding and decoding video. This “one size fits all” paradigm is a compromise between the requirements of multiple types of video content (sports, movies, conversational, surveillance, etc), multiple applications and multiple platforms. The changing nature of the video communication landscape and the proliferation of applications, platforms and coded formats create significant challenges:

(i) There is a long timescale between proposing a new idea or concept and implementing this concept in practical devices and systems. A new video coding algorithm must typically be adopted in a standard or de-facto standard before it can be brought to market, a slow and expensive process. Video coding standards themselves are becoming increasingly complex and involve a massive technical effort from hundreds of contributors.

(ii) Fixed, standardized approaches to video coding are not capable of keeping pace with the increasing variety of video coding, transmission and decoding scenarios, leading to sub-optimal coding performance and user experiences. A standards-based codec is inherently limited in its ability to adapt to the characteristics of the video source and/or application scenario. In contrast, dynamically adaptive coding algorithms have the potential to significantly improve compression performance.

(iii) There is an increasing requirement for decoding platforms to support multiple coding standards. For example, a Blu-Ray player must support MPEG-2, VC-1 and H.264 video coding standards. With no straightforward method of re-using functionality across these formats, this leads to over-designed decoders. The problem grows as new standards are released and legacy formats continue to be supported.

Adaptive and/or configurable video coding has the potential to address the problems listed above. Some steps have already been taken towards increasing the flexibility of video coding methods. In this paper we describe emerging platforms for configurable coding and demonstrate the benefits of a more flexible approach to video compression.

## II. CONFIGURATION APPROACHES

### A. MPEG Reconfigurable Video Coding

The MPEG Reconfigurable Video Coding initiative, RVC, aims “to provide a framework allowing a dynamic development, implementation and adoption of standardized video coding solutions with features of higher flexibility and reusability” [1]. RVC is motivated by the recognition that standards frameworks need to evolve in order to efficiently support multiple codec configurations and to facilitate innovation in codec design.

MPEG RVC is developing two standards, Codec Configuration Representation [2] and Video Tool Library [3], due for completion in 2008/09. The Video Tool Library specifies a set of Functional Units (FUs), “building blocks” for video decoders,

such as transforms, motion compensators and entropy decoders. The structure of a video decoder (composed of selected FUs and their interconnections) is defined by a Decoder Description Language (DDL) and the format of the coded bitstream is defined using Bitstream Syntax Description Language (BSDL), both specified in the Codec Configuration Representation standard.

**Figure 1** illustrates a typical RVC decoding scenario. In order to decode a video bitstream, the decoder needs to know (a) how to parse the bitstream and extract the coded data elements and (b) how to decode these elements. The RVC decoding engine receives BSDL and DDL specifications in compressed form. The decoder composition module generates a decoding solution (an actual video decoder) based on the BSDL and DDL specifications. It makes use of selected FUs from the Video Tool Library and connects these according to the DDL. Once the decoding solution has been generated, it can then decode the video bitstream.

This approach has a number of potential benefits. A decoder can be modified to decode a different format by sending new BSDL/DDDL descriptions, enabling efficient support for multiple coding formats. A non-standard coding format can be supported provided it uses FUs available to the decoder (i.e. FUs in the decoder's VTL). In practical terms this means that a new format should use FUs standardized by MPEG. A new coding tool can be proposed to MPEG for

standardization as a new FU – a potentially faster, simpler route than creating a complete new video coding standard.

However, the RVC framework has some limitations. Innovation in decoder design is restricted to the existing set of FUs. There is still likely to be a significant time lag between proposing a new coding tool, standardizing it as a new FU and propagating this to all decoders in an updated VTL. At present, RVC does not envision dynamic re-configuration, limiting the ability to adapt to the statistics of the current video scene.

### B. Adaptive Video Coding

There is significant illustration in the literature of the benefits of adaptive processing in image/video coding, in which elements of the compression process are modified during coding, depending on the statistics of the source data. A well-known example is the Karhunen-Loeve transform (KLT) in which the transform bases depend on the current data. While there have been arguments against the KLT for lossy compression of non-Gaussian sources [4], this approach has been made practical in a variety of ways including exclusive use of the KLT [5,6] and through adaptive switching between the DCT and the KLT [7]. Swapping between pre-defined coding elements can give a rate-distortion benefit [8]. However, at present there are no frameworks to enable implementation of these or related adaptive

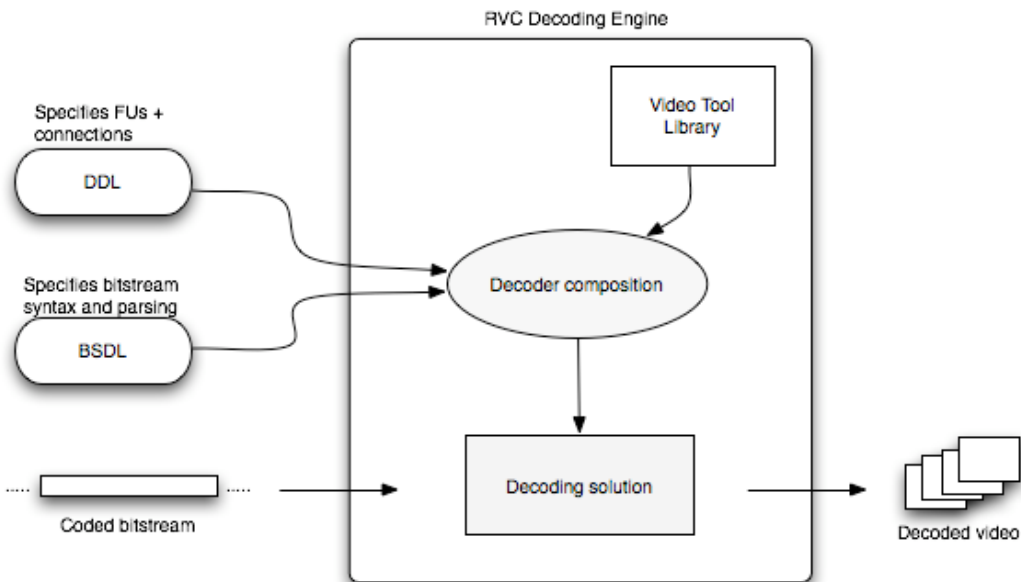


Figure 1 MPEG RVC decoding scenario

techniques in practical video coding applications. MPEG RVC is not (at present) intended to support dynamic re-configuration (modifying the decoder structure and/or tools during coding) and so a new approach is needed to put adaptive coding into practice.

### C. Dynamically Configurable Video Coding

We propose an alternative (or potential extension) to the RVC model in which a common decoding engine (Universal Video Decoder, UVD) can be configured to decode any video sequence or syntax. In contrast to RVC, our framework supports (a) complete re-configuration of individual coding tools and (b) dynamic re-configuration during coding.

In a typical scenario, the UVD initially has no knowledge of the decoding methods required for a particular video bitstream. The encoder sends a set of configuration commands (Decoder Description Syntax, DDS) and the UVD generates and connects new functional processing units according to these commands. The UVD can then proceed to decode the video bitstream. At a later point, the encoder may signal a change in configuration by sending new Decoder Description Syntax; the UVD implements the change and continues to decode the bitstream using the changed syntax or functionality.

This approach has the following benefits:

(i) Any video decoding processes may be created (instantiated) without the need for a standardized library of processes. Hence the time

lag between developing a new algorithm and propagating it to existing decoders becomes very short.

(ii) Re-configuration may be carried out dynamically, enabling on-the-fly adaptation. This enables the codec to use the best configuration for a particular video stream, with significant potential for improved compression performance.

(iii) A single UVD platform can be re-configured to support multiple standard or non-standard video compression formats.

## III. DYNAMIC CONFIGURATION IN PRACTICE

### A. Prototype Dynamically Configurable Codec

Figure 2 illustrates our prototype dynamically configurable video coding framework. Decoding functions and their interconnections are defined in a high-level language (a subset of C). This is parsed (converted) into a machine-readable object code form. This decoder description is encoded and transmitted as Decoder Description Syntax (DDS). The UVD decodes the DDS and creates the necessary decoding functions.

The prototype UVD operates as follows. Each decoding function is instantiated as a software object with its own data and “program” memory. The program memory contains a sequence of opcodes, instructions for video and data processing. The set of opcodes is designed as a compromise between flexibility and performance, with the aim of supporting a wide range of processing functions without compromising

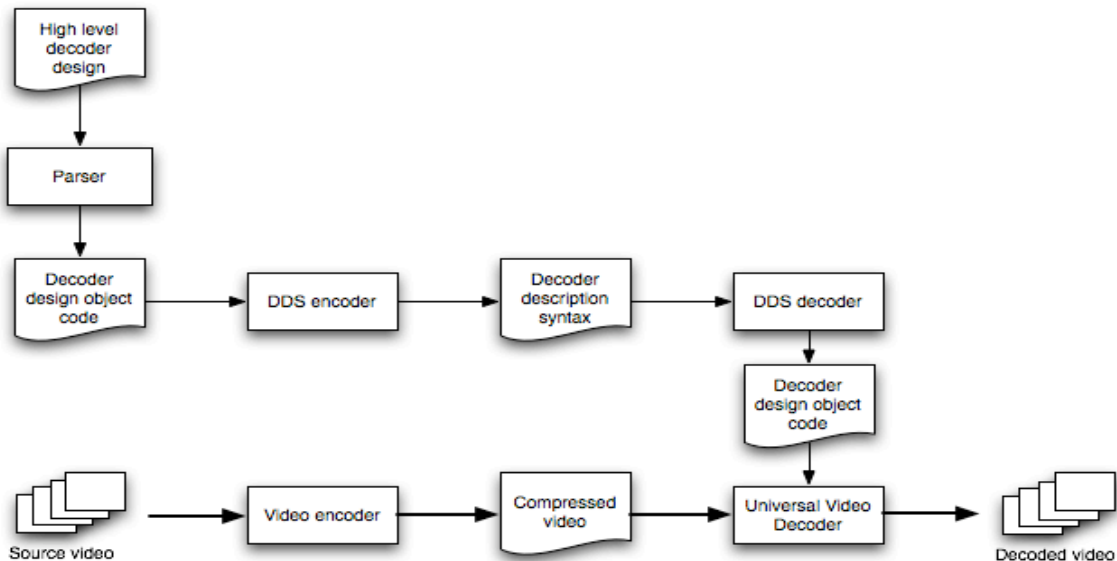


Figure 2 Dynamic Configuration Framework

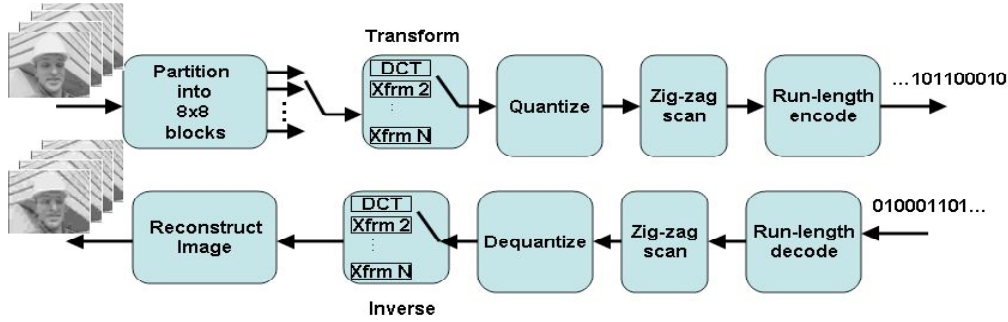


Figure 3 Simple intraframe coder with transform reconfiguration

computational efficiency. The UVD acts as a virtual processor, stepping through each decoding function object and branching between objects as required, to decode compressed video.

Configuration and re-configuration of the UVD is achieved by sending decoding objects, replacing existing objects (e.g. replacing one transform with another) or creating new objects (which are linked in to the current decoder structure). In our prototype system, re-configuration may occur once per frame.

**B. Performance Results**

We present selected performance results for a simple intraframe codec (Figure 3). For a detailed overview of this experiment and an analysis of a similar approach in the H.264 framework, please see [9,10]. Each frame is partitioned into 8x8 blocks which are transformed, quantized, re-ordered and run-length coded. We compare three configurations:

(A) The decoder uses a single transform, the DCT, for every block. Quantizer step size is fixed throughout the sequence.

(B) The decoder uses two transforms, DCT and Haar. The encoder chooses the best transform for each block to minimize the rate for a given quantizer step size. The decoder design is fixed, i.e. we assume prior knowledge of the available transforms.

(C) Reconfigurable decoder. The decoder is initially configured as per Codec A, using only the DCT. A description of an inverse Haar transform function is coded and sent, after which the decoder may use either transform (as per Codec B).

Rate-distortion curves for this experiment are shown in Figure 4 for the “Carphone” video sequence (QCIF, 100 frames). Codec B (with the DCT and Haar transforms as a fixed configuration)

has the best rate-distortion performance, due to its ability to choose the best transform to suit the data in each image block. However, Codec C (with the Haar sent dynamically as a new configuration) performs nearly as well as Codec B, despite the extra bitrate required to send the new coded configuration. Codec C has the significant advantage that any new transform may be sent without prior knowledge by the decoder.

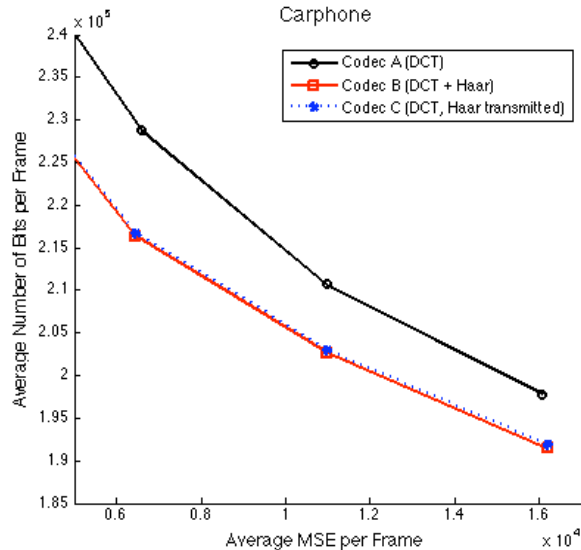


Figure 4 Rate-distortion performance (Carphone)

Table 1 compares the processing resources required by “fixed” and reconfigurable versions of the two transforms (inverse DCT and inverse Haar). The fixed version is implemented as a standard C function whereas the reconfigurable version is implemented as a configurable decoding object (see section III.A). The Table records the time taken to carry out 1 million transforms on an Intel P4 processor with 512MB RAM. No processor-specific optimizations were used. The reconfigurable transform is slower in each case, but the performance penalty is modest given the

significant improvement in flexibility offered by this approach.

Table 1 Comparison of processing resources

Transform	Fixed	Reconfigurable	Ratio
DCT	2.890 s	7.859 s	2.62
Haar	2.235 s	6.594 s	2.95

#### IV. FUTURE CHALLENGES

The MPEG RVC standards are due to be published within months. The Video Tool Library [3] includes Functional Units required to construct popular standard decoders such as MPEG-2 and H.264/AVC. With suitable BSDL and DDL descriptions, an RVC-compatible decoder may be configured into any of the current popular standards, with the potential for increased flexibility and efficiency. RVC necessitates a substantially new approach to hardware and software design of decoders [11] and it will be necessary to demonstrate the benefits of RVC in order to convince manufacturers to adopt this approach. As new video coding methods are developed, it will be important to streamline the standardization of new FUs and the propagation of these to RVC decoders.

Our proposed framework for dynamically configurable video coding has significant potential benefits for future video codec implementations. We have developed a prototype system that demonstrates dynamic re-configuration in a real time software codec, showing that the concept is feasible. However, a number of challenges need to be addressed in order to take this work forward, including:

1. Demonstrating the capability to describe a wide range of decoding functionality, without compromising computational performance and decoder memory requirements.
2. Developing and implementing practical algorithms to exploit the rate-distortion benefits of adaptive coding.
3. Defining the framework, including a complete specification for DDS and reference UVD implementations.
4. Developing reference designs for standards-based decoders in DDS as a starting point for implementing current and emerging codec designs.

5. Addressing practical issues such as error resilience, synchronisation between encoder and decoder configurations and working within the memory and computation constraints of the decoder platform.

Notwithstanding these challenges, configurable coding is an ambitious new approach to video compression that opens up a range of research opportunities and has the potential to dramatically change the landscape of video coding research, development and implementation.

Further information about dynamically configurable video coding:

<http://sourceforge.net/projects/reconfigvid/>

#### REFERENCES

- 1 E. S. Jang, J. Ohm and M. Mattavelli, "Whitepaper on Reconfigurable Video Coding (RVC)", ISO/IEC JTC1/SC29/WG11 document N9586, Antalya, January 2008.
- 2 ISO/IEC 23001-4 "Information technology – MPEG systems technologies – Part 4: Codec configuration representation", Committee Draft, October 2007.
- 3 ISO/IEC 23002-4 "Information technology – MPEG video technologies – Part 4: Video tool library", Committee Draft, October 2007.
- 4 M Effros, H Feng and K Zeger, "Suboptimality of the Karhunen-Loeve transform for transform coding", *IEEE Trans. Information Theory*, vol. 50, no. 8, pp. 1605-1619, August 2004.
- 5 R Dony and S Haykin, "Optimally adaptive transform coding", *IEEE Trans. Image Processing*, vol. 4, no. 10, October 1995.
- 6 C Archer and T K Leen, "A generalized Lloyd-type algorithm for adaptive transform coding", *IEEE Trans. Image Processing*, vol. 52, no. 1, January 2004.
- 7 A Dapena and S Ahalt, "A hybrid DCT-SVD image coding algorithm", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 12, no. 2, February 2002.
- 8 K Zhang and J Kittler, "Framework for dynamically reconfigurable video codec using multiple coding tools", Proc. SPIE vol. 3408, *Broadband European Networks and Multimedia Services*, September 1998.
- 9 S Kannangara, I Richardson, M Bystrom and M de Frutos Lopez, "Fast, dynamic configuration of transforms for video coding", Proc. International Conference on Image Processing, San Diego, October 2008.
- 10 I Richardson, M Bystrom, M de Frutos Lopez and S Kannangara, "Dynamic transform replacement in an H.264 codec", Proc. International Conference on Image Processing, San Diego, October 2008.
- 11 J-M Hsiao, C-J Tsai, "Analysis of an SOC architecture for MPEG reconfigurable video coding framework", Proc. IEEE ISCAS, May 2007.