Design of Compressed Memory Model Based on AVC Standard for Robotics

Devaraj Verma C[1] and Dr. VijayaKumar M.V[2]

[1] Lecturer, Department of Information Science and Engineering, PES Institute of Technology, 100-ft Ring Road, BSK-III stage, Bangalore-560085, Karnataka-INDIA

[2] Prof and Head of Department of Information Science and Engineering, New Horizon College of Engineering, Panathur post, marathahalli, Bangalore, Karnataka-INDIA

Abstract

To cater the needs of diverse application domains, three basic feature sets called profiles are established in H.264 standard: the Baseline, Main, and Extended profiles. The Baseline profile is designed to minimize complexity and provide high robustness and flexibility for use over a broad range of network environments and conditions; the Main profile is designed with an emphasis on compression coding efficiency capability; and the Extended profile is designed to combine the robustness of the Baseline profile with a higher degree of coding efficiency and greater network robustness. The H.264 bitstream is organized as a series of NAL units, which are first, processed by the NAL unit decoder module. NAL unit decoder does the job of NAL unit separation and parsing of the header information. The output data elements of NAL unit decoder are entropy decoded and reordered to produce a set of quantized coefficients by the variable length decoder module. The quantized co-efficients are then rescaled and inverse transformed to give difference macroblock using the transformation and quantization module. Using the header information decoded from the bitstream, the motion compensation and picture construction module produces distorted macroblocks which are then filtered using the deblocking filter to create decoded macroblocks. The output is stored as a yuv file, which can be played using any YUV player/viewer. Thus the visual input from the sensors of the robots can be compressed by this technique before delivered to the actuators on the scene.

KeyWords: *Deblocking filter, Motion Compensation, Macroblock modes, NAL units*

1. Introduction

H.264 is a new video compression scheme that is becoming the worldwide digital video standard for consumer electronic and personal computers [1,2,3,4,5].Uncompressed multimedia (graphics, audio and video) data requires considerable storage capacity and transmission bandwidth. Despite rapid progress in mass-storage density, processor speeds, and digital communication system performance, demand for data storage capacity and data-transmission bandwidth continues to outstrip the capabilities of available technologies [6,7,8,9,10]. ITU-T H.264 / MPEG-4 (Part 10) Advanced Video Coding (commonly referred as H.264/AVC) is the latest entry in the series of international video coding standards [11,12,13,14]. It is currently the most powerful and state-of-the-art standard developed by a Joint Video Team (JVT) consisting of experts from ITU-T's Video Coding Experts Group (VCEG) and ISO/IEC's Moving Picture Experts Group (MPEG)[15,16,17]. With the wide breadth of applications considered by the two organizations, the application focus for the work was correspondingly broad – from video conferencing to entertainment (broadcasting over cable, satellite, terrestrial, cable modem, DSL etc.; storage on DVDs and hard disks; video on demand etc.) to streaming video, surveillance and military applications, and digital cinema. Three basic feature sets called profiles are established to address these application domains: the Baseline, Main, and Extended profiles as shown in Fig 1.

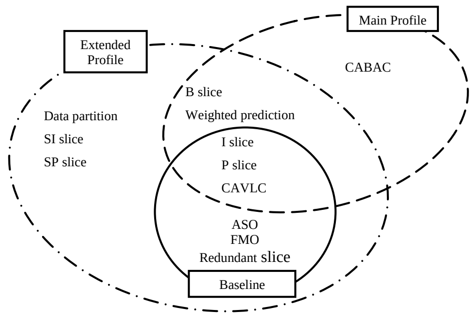


Fig 1 H.264 Profiles

Baseline is a low-complexity low-latency profile, which provides basic functionality. Typical applications are interactive ones like mobile video telephony and video conferencing. The Main profile targets studio, broadcast, and other high-quality applications like HDTV and DVD. It is a high-complexity high-latency profile without error-resilient features. Finally, Extended is highly error-robust and of low complexity and may be used in e.g. streaming applications. All profiles will have the Baseline compatibility.

2. NAL units, slices, fields and frames

A H.264 video stream is organized in discrete packets, called "NAL units" (Network Abstraction Layer units). Each of these packets can contain a part of a slice, that is, there may be one or more NAL units per slice. But not all NAL units contain slice data; there are also NAL unit types for other purposes, such as signaling, headers and additional data. The slices, in turn, contain a part of a video frame. In normal bitstreams, each frame consists of a single slice whose data is stored in a single NAL unit. The possibility to spread frames over an almost arbitrary number of NAL units can be useful if the stream is transmitted over an error-prone medium: The decoder may resynchronize after each NAL unit instead of skipping a whole frame if a single error occurs.

H.264 also supports optional interlaced encoding. In this encoding mode, a frame is split into two fields. Fields may be encoded using spatial or temporal interleaving. To encode color images, H.264 uses the YCbCr color space like its predecessors, separating the image into luminance (or "luma", brightness) and chrominance (or "chroma", color) planes. It is, however, fixed at 4:2:0 sub sampling, the chroma channels each have half the resolution of the luma channels.

Slice types
H.264 defines five different slice types: I, P, B, SI and SP.

I slices or "Intra" slices describe a full still image, containing only references to itself. A video stream may consist of only I slices, but this is typically not used. The first frame of a sequence always needs to be built out of I slices.
P slices or "Predicted" slices use one or more recently decoded slices as a reference (or "prediction") for picture construction. The prediction is usually not exactly the same as the actual picture content, so a "residual" may be added.

B slices or "Bi-Directional Predicted" slices work like P slices with the exception that former and future I or P slices (in playback order) may be used as reference pictures. For this to work, B slices must be decoded after the following I or P slices.

SI and **SP** slices or "Switching" slices may be used for transitions between two different H.264 video streams. This is a very uncommon feature.

3. Motion Compensation

Since MPEG-1, motion compensation is a standard coding tool for video compression. Using motion compensation, motion between frames can be encoded in a very efficient manner. A typical P-type block copies an area of the last decoded frame into the current frame buffer to serve as a prediction. If this block is assigned a nonzero motion vector, the source area for this copy process will not be the same as the destination area. It will be moved by some pixels, allowing to accommodate for the motion of the object that occupies that block. Motion vectors need not be integer values: In H.264, motion vector precision is one-quarter pixel (one-eighth pixel in chroma). Interpolation is used to determine the intensity values at non-integer pixel positions. Additionally, motion vectors may point to regions outside of the image. In this case, edge pixels are repeated.

Because adjacent blocks tend to move in the same directions, the motion vectors are also encoded using prediction. When a block's motion vector is encoded, the surrounding blocks motion vectors are used to estimate the current motion vector. Then, only the difference between this prediction and the actual vector is stored.

4. Macroblock Layer

Each slice consists of macroblocks (or, when using interlaced encoding, macroblock pairs) of 16x16 pixels. The encoder may choose between a multitude of encoding modes for each macroblock.

4.1 Macroblock modes for I slices

In H.264, I slice uses a prediction/residual scheme: Already decoded macroblocks of the same frame may be used as references for intra prediction process. The macroblock mode indicates which of two possible prediction types is used:

Intra_16x16 uses one intra prediction scheme for the whole macroblock. Pixels may be filled from surrounding macroblocks at the left and the upper edge using one of four possible prediction modes. Intra prediction is also performed for the chroma planes using the same range of prediction modes. However, different modes may be selected for luma and chroma.

Intra_4x4 subdivides the macroblock into 16 sub blocks and assigns one of nine prediction modes to each of these 4x4 blocks. The prediction modes offered in Intra_4x4 blocks support gradients or other smooth structures that run in one of eight distinct directions. One additional mode fills a whole sub block with a single value and is used if no other mode fits the actual pixel data inside a block. Intra chroma prediction is performed in an identical way to the one used in Intra_16x16 prediction. Thus, the chroma predictions are not split into sub blocks as it is done for luma. Additionally, a third intra macroblock type exists: I_PCM macroblocks contain raw uncompressed pixel data for the luma and chroma planes. This macroblock type is normally not used.

4.2 Macroblock modes for P and B slices

P and B slices use another range of macroblock modes, but the encoder may use intra coded macroblocks in P or B slices as well. In regions of uniform motion without texture changes, macroblocks may be skipped. In this case, no further data is stored for the macroblock, but it is motion-compensated using the predicted motion vector. Otherwise, the macroblock of 16x16 pixels is divided into macroblock partitions:

- It may be stored as a single partition of 16x16 pixels.
- It may be split horizontally into two partitions of 16x8 pixels each.
- It may be split vertically into two partitions of 8x16 pixels each.
- It may be split in both directions, resulting in four sub-macroblock partitions of 8x8 pixels.

Each of these may be split similarly into partitions of 8x8, 8x4, 4x8 or 4x4 pixels. Not all sub-macroblock partitions need to be split in the same manner, thus allowing for any number of partitions from 1 to 16.
Each macroblock or sub-macroblock partition stores its own motion vector, thus allowing for motion isolation of objects as small as 4x4 pixels. Additionally, a reference frame number that is used for prediction is stored for each partition.

5. Block Transformation and Encoding

The basic image encoding algorithm of H.264 uses a separable transformation. The mode of operation is similar to that of JPEG and MPEG, but the transformation used is not an 8x8 DCT, but a 4x4 integer transformation derived from the DCT. This transformation is very simple and fast; it can be computed using only additions/subtractions and binary shifts. It decomposes the image into its spatial frequency components like the DCT, but due to its smaller size, it is not as prone to high frequency “mosquito” artifacts as its predecessors.

An image block *B* is transformed to *B*’ using the formula1. The necessary post-scaling step is integrated into quantization and therefore omitted:

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{pmatrix}$$

$$B' = MBM^T \qquad \text{………(1)}$$

The basic functionality of the H.264 image transformation process is as follows: For each block, the actual image data is subtracted from the prediction. The resulting residual is transformed. The coefficients of this transform are divided by a constant integer number. This procedure is called quantization; it is the only step in the whole encoding process that is actually lossy. The divisor used is called the quantization parameter; different quantization parameters are used for luma and chroma channels. The quantized coefficients are then read out from the 4x4 coefficient matrix into a single 16-element scan. This scan is then encoded using sophisticated (lossless) entropy coding. In the decoder, these steps are performed in reversed order.

5.1 Entropy encoding modes

H.264 supports two different methods for the final entropy encoding step: CAVLC, or Context-Adaptive Variable Length Coding, is the standard method using simple variable length Huffman-like codes and codebooks. CABAC, or Context-

Adaptive Binary Arithmetic Coding, on the other side, is an optional, highly efficient binary encoding scheme.

5.2 DC transformation

The upper left transform coefficient (i.e. the first coefficient in scan order) is treated separately for Intra_16x16 macroblocks and chroma residuals. Because this coefficient indicates the average intensity value of a block, correlations between the DC values of adjacent equally predicted blocks can be exploited this way.For Intra_16x16 macroblocks, the DC coefficients are transformed using a separable 4x4 Hadamard transform with the following matrix:

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \qquad \ldots\ldots\ldots\ldots(2)$$

Chroma residuals are always transformed in one group per macroblock. Thus, there are 4 chroma blocks per macroblock and channel. A separable 2x2 transform is used:

$$M = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \qquad \ldots\ldots\ldots\ldots(3)$$

6. Advanced Coding Tools

Beyond the basic structure described above, H.264 offers additional features to improve coding efficiency.

6.1 In-Loop Deblocking Filter

After each individual frame has been decoded, a deblocking filter that reduces the most visible blocking artifacts is applied. This has been available since MPEG-4 Simple Profile as an optional post-processing operation, but in H.264, it is closely integrated into the encoding and decoding process: The already deblocked frames are used as reference frames by the following P or B slices. This technique circumvents noticeable blocking artifacts as far as possible.

6.2 Advanced prediction

H.264 may not only use interframe references to the last decoded frame, but to an arbitrary number of frames. This greatly improves the encoding efficiency of periodic movements (long-term prediction). Moreover, multiple predictions may be mixed by arbitrary ratios (weighted prediction).

6.3 Arbitrary Slice Ordering

Since the slices of a picture can be decoded independently (except for the slice edges which can only be approximated), slices need not be decoded in the correct order to render an image in acceptable quality. This is useful e.g. for UDP streaming where packets may be delivered out-of-order.

6.4 Flexible Macroblock Ordering

The macroblocks inside a slice may be encoded in any order. This can be used to increase robustness against transmission errors, for example. It is also reminiscent of MPEG-4's "video object planes" system, which could be used to encode each object of a scene individually. In normal video streams, however, this feature is not used, and macroblocks are sent in the normal scan line order.

7. SYSTEM DESIGN

7.1 Introduction

Design is the first step in moving from problem domain towards the solution domain. A design can be object oriented or function oriented. In this paper function oriented method is followed which consists of module definition with each module supporting a function abstraction. As shown in the below Fig2 sensor image output can be compressed by the H.264 standard before delivered to the actuators on the scene.

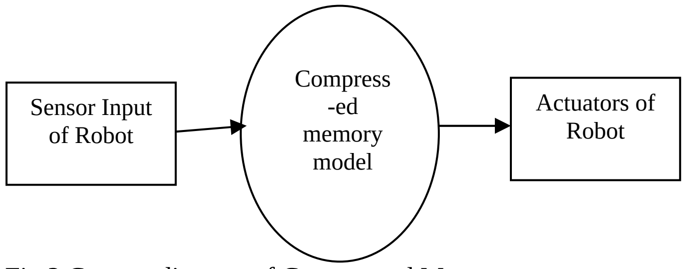


Fig 2 Context diagram of Compressed Memory model for Robotics

7.2 Structured design

The context diagram for the H.264 Baseline decoder is as shown in Fig 3. The input and output of the system are as given in this diagram.

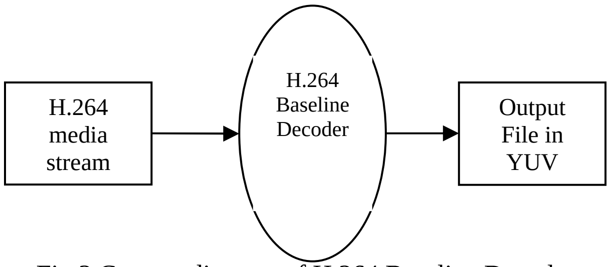


Fig 3 Context diagram of H.264 Baseline Decoder

The input to the decoder is a video file/bitstream, which is encoded using H.264 standard. The decoder decodes the input file and stores the output as a yuv file. The output file can be played using any YUV Player/Viewer.Using this as the starting point, a logical data flow diagram of the physical system is constructed. The DFD (Data Flow Diagram) for the same is shown Fig 4.

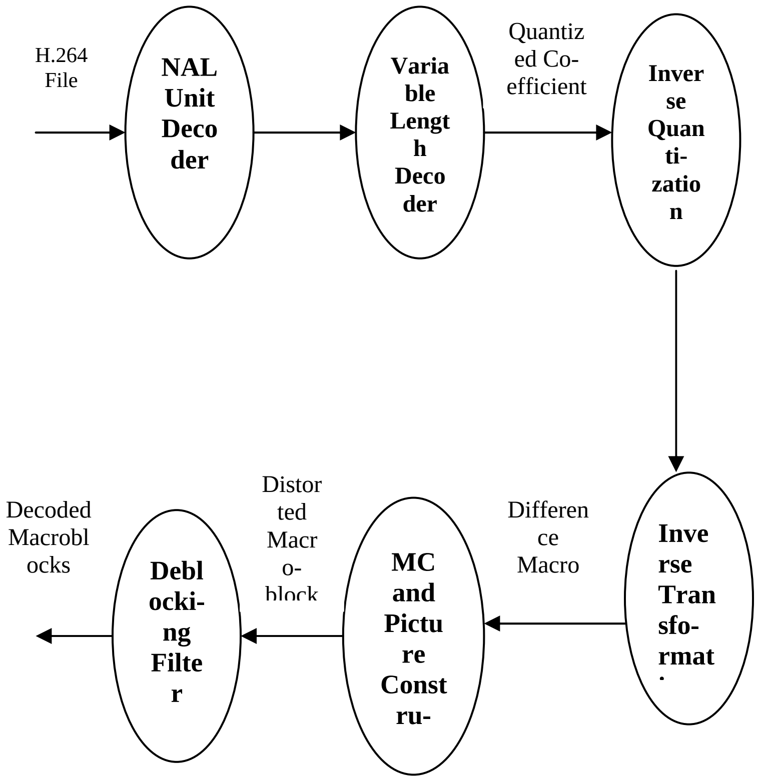


Fig 4: Logical DFD of H.264 Baseline Decoder

As shown in the above DFD, the system designed is desired to have four major functional modules namely NAL Unit Decoder, Variable Length Decoder, Inverse transformation and Quantization module and Motion Compensation and Picture construction module, which will perform the following processes to obtain the decoded video frames from the encoded H.264 Byte stream.

1 NAL unit decoding process
2 Variable length decoding process
3 Inverse transformation and quantization process
4 Motion compensation and picture construction process

Apart from these four main functional blocks, a deblocking filter is also needed in the system which filters the edges of the distorted macroblocks; thus helping in eliminating blocking artifacts and improving video quality.

7.3 Functional and Design specification

Here, the major requirements are listed and then a list of modules in the system that implements those requirements is given.

1 A user friendly GUI should be provided for easy interaction with the decoder.

2 The input to the decoder should be a media file encoded using H.264 baseline profile.

3 The validity of the input file should be checked.

4 For bitstreams which do not confirm to H.264 baseline profile an error message should be flashed to the user.

5 Four processes NAL unit decoding, Variable length decoding, Inverse transformation and quantization and Motion compensation and picture construction should be performed on the encoded media file to get decoded video frames.

6 Edges of the decoded macroblocks/frames should be filtered out to eliminate blocking artifacts and give an impression of finer quality video.

1 A status file and a log file should be generated to review the decoding process.

2 The decoded output should be stored as a yuv file.

NAL Unit Decoder module:
Input:
H.264 bitstream.

Output:
NAL unit decoded data.
Purpose:
It is the first stage of the decoder. The input to the decoder is a H.264 bitstream which is organized as discrete packets called NAL units of variable length. This stage is used for separating the NAL units and for processing the header information contained in these NAL units. Using the header information this module also checks the validity of the input bitstream.

Variable Length Decoder module:
Input:
Data from NAL unit Decoder.
Output:
Quantized co-efficients.
Purpose:
In this stage the data elements from the NAL unit decoder are entropy decoded and reordered to produce a set of quantized coefficients.

Inverse Transformation and Quantization module:
Input:
Quantized co-efficients.
Output:
Difference macroblock.
Purpose:
In this stage the quantized co-efficients from the variable length decoding unit are rescaled and inverse transformed to give difference macroblock.

Motion Compensation and Picture Construction module:
Input:
Difference macroblock.

Output:
Distorted macroblock.
Purpose:
In this stage the difference macroblock is taken as an input and distorted macroblock is produced as output. It is here in this stage that the actual picture construction takes place.

Deblocking Filter module:
Input:
Distorted macroblock.
Output:
Decoded Macroblock.
Purpose:
The purpose of this stage is to filter the edges of the macroblocks/frames to eliminate blocking artifacts and give an impression of finer quality video.

8. Data Flow Diagrams

In this section flow charts are given for the NAL Unit decoder module, Variable length decoder module, Inverse transformation and quantization module and Motion compensation and picture construction module. Also a flow control diagram indicating decoder control flow is given.

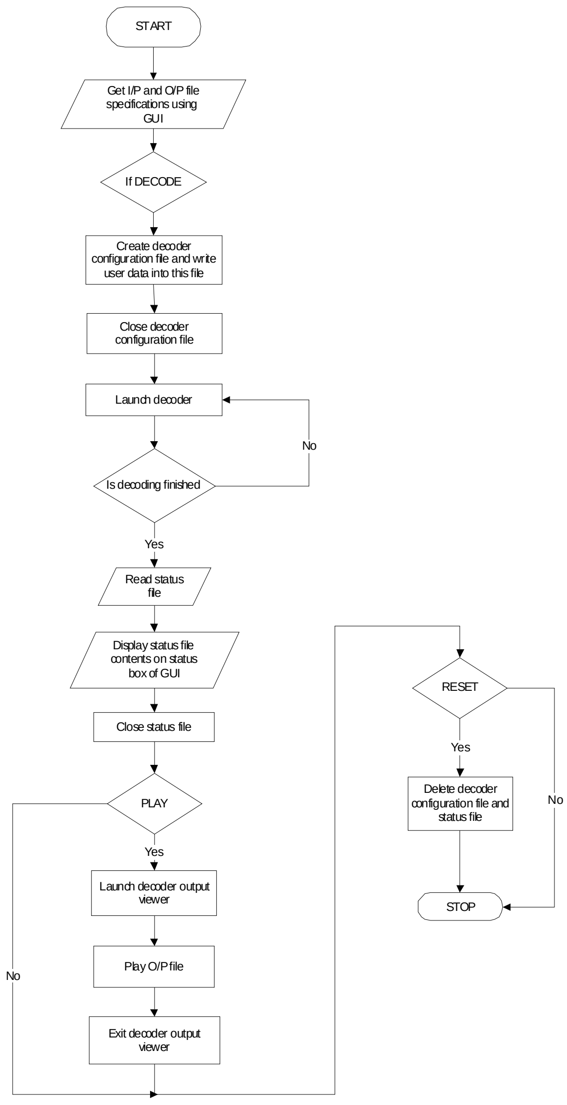

START
Get I/P and O/P file specifications using GUI
If DECODE
Create decoder configuration file and write user data into this file
Close decoder configuration file
Launch decoder
No
Is decoding finished
Yes
Read status file
Display status file contents on status box of GUI
Close status file
PLAY
Yes
Launch decoder output viewer
Play O/P file
Exit decoder output viewer
No
RESET
Yes
Delete decoder configuration file and status file
No
STOP

Fig 5: Decoder Control flow

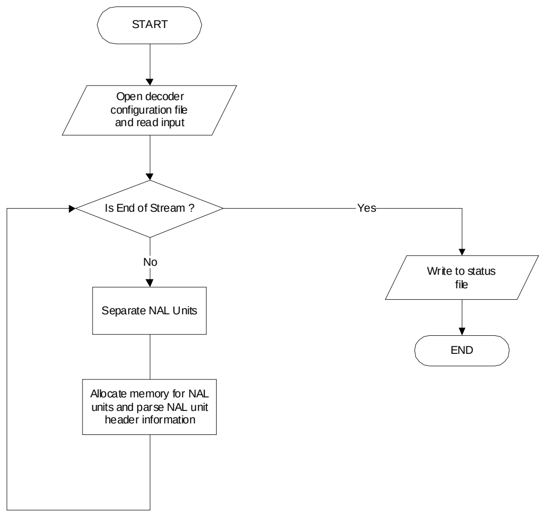

START
Open decoder configuration file and read input
Is End of Stream ?
Yes
No
Separate NAL Units
Allocate memory for NAL units and parse NAL unit header information
Write to status file
END

Fig 6: NAL Unit Decoder module

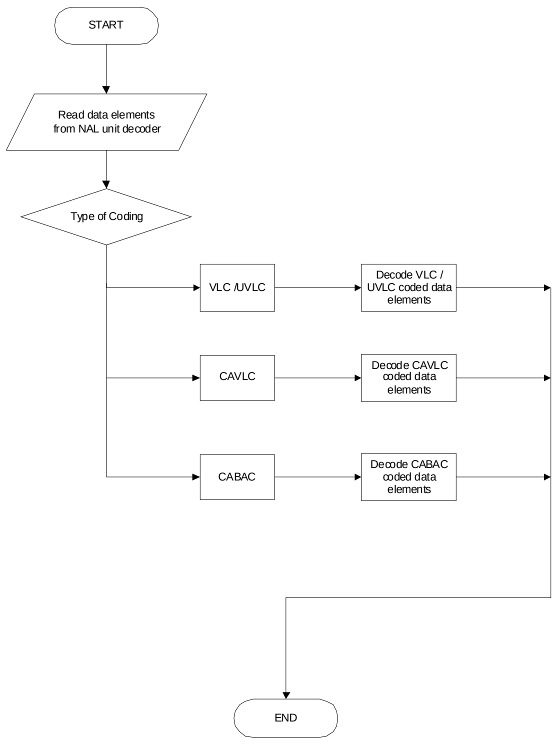

START
Read data elements from NAL unit decoder
Type of Coding
VLC /UVLC
Decode VLC / UVLC coded data elements
CAVLC
Decode CAVLC coded data elements
CABAC
Decode CABAC coded data elements
END

Fig 7: Variable Length Decoder module

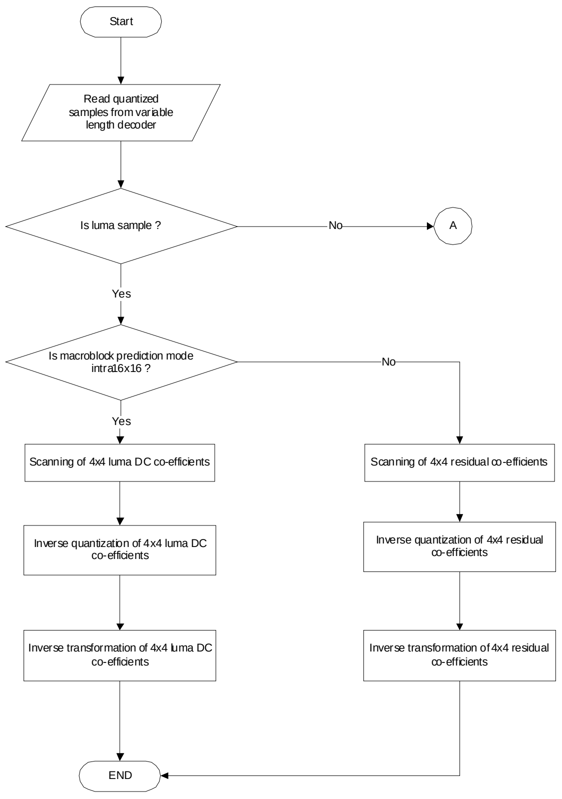
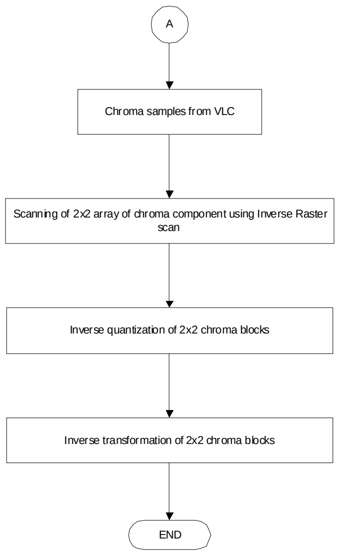

Start
Read quantized samples from variable length decoder
Is luma sample ?
No
A
Yes
Is macroblock prediction mode intra16x16 ?
No
Yes
Scanning of 4x4 luma DC co-efficients
Scanning of 4x4 residual co-efficients
Inverse quantization of 4x4 luma DC co-efficients
Inverse quantization of 4x4 residual co-efficients
Inverse transformation of 4x4 luma DC co-efficients
Inverse transformation of 4x4 residual co-efficients
END

Fig 8: Inverse transformation and quantization module


A
Chroma samples from VLC
Scanning of 2x2 array of chroma component using Inverse Raster scan
Inverse quantization of 2x2 chroma blocks
Inverse transformation of 2x2 chroma blocks
END

Inverse transformation and quantization module (Cont'd)

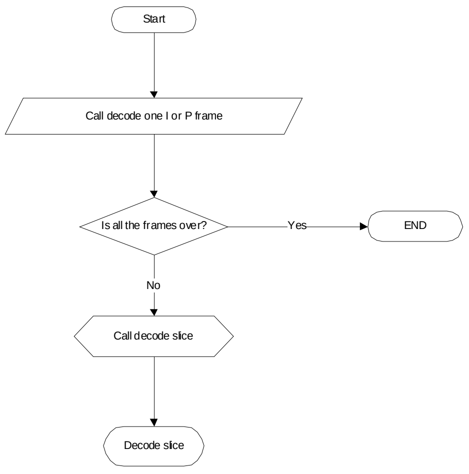
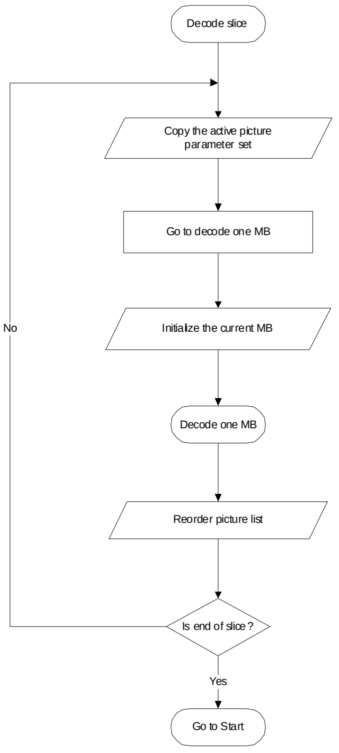
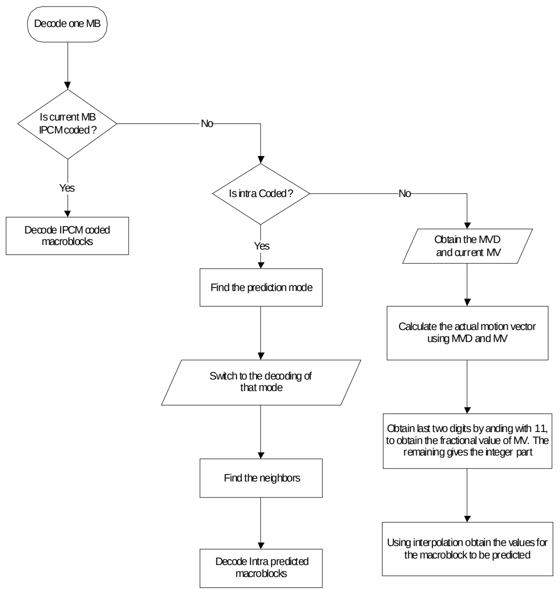


Fig 9: Motion Compensation and Picture Construction module



Motion Compensation and Picture Construction module (Cont'd)


Decode one MB
Is current MB IPCM coded ?
No
Yes
Decode IPCM coded macroblocks
Is intra Coded ?
No
Yes
Obtain the MVD and current MV
Find the prediction mode
Calculate the actual motion vector using MVD and MV
Switch to the decoding of that mode
Obtain last two digits by anding with 11, to obtain the fractional value of MV. The remaining gives the integer part
Find the neighbors
Using interpolation obtain the values for the macroblock to be predicted
Decode Intra predicted macroblocks

Motion Compensation and Picture Construction (Cont'd)

9. Conclusion

H.264 standard is found to increase coding efficiency compared to any previous video coding standards (like MPEG-4 part 2 and MPEG-2) for a wide range of bit rates and resolutions. Thus the visual input from the sensors of the robots can be compressed by this technique before delivered to the actuators on the scene.

H.264 because of its higher compression is definitely going to be the prominent standard in the near future. It will make possible new applications that could have never been dreamed of before. The higher performance of the H.264 is mainly due to the tree structured motion compensation, which has also made the implementation of the standard simpler that many previous standards.

The increased compression performance of H.264 will enable existing applications like video conferencing, streaming video over the internet, and digital television on satellite and cable to offer better quality video at lower cost, and will allow new video applications that were previously impractical because of economics or technology. High-Definition television on DVD, Video on mobile phones and video conferencing over low bandwidth connections will become practical and it is believed that important new applications will be invented that will leverage H.264 technology.

References

[1] A.K.Jain. "Fundamentals of Digital Image Processing". Englewood Cliffs, NJ: Prentice Hall, 1989
[2] Peter Symes, "Digital Video Compression", McGraw-Hill, 2003.
[3]Khalid Sayood. "Introduction to Data Compression", second edition
[4] William B. Pennebaker and Joan L. Mitchell, "JPEG Still Image Data Compression Standard", Van Nostrand Reinhold, 1993.
[5] Zi Nan Li and Mark S Drew, "Fundamentals of multimedia", Pearson Publication.
[6] Joan L. Mitchell, "MPEG Video Compression Standard", Chapman & Hall: International Thomson, 1996.
[7 MPEG Video Compression Technique, http://rnvs.informatik.tuchemnitz.de/~jan/MPEG/HTML/mpeg_tech.html .
[8] MPEG website: http://www.mpeg.org
[9] N .Abramson. "Information theory and coding". New York: McGraw-Hill, 1963
[10] T.A. Welch. "A Technique for High-Performance data compression", IEEE Computer, pages 8-19, June 1984
[11] M. Ghanbari, "Standard Codecs: Image Compression to Advanced Video Coding," Hertz, UK: IEE, 2003.
[12] MPEG-2: ISO/IEC JTC1/SC29/WG11 and ITU-T, "ISO/IEC 13818-2: Information Technology-Generic Coding of Moving Pictures and Associated Audio Information: Video," ISO/IEC and ITU-T, 1994.
[13] IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on the H.264/AVC Video Coding Standard, Vol. 13, No. 7, July 2003.
[14] P. Hoyingcharoen and C. Schmidt, "Overview of H.264/AVC, Signal Compression Lab, UCSB".
[15] ISO_IEC_14496-10 first edition December 2003.
[16] Iain E. G. Richardson. H.264 / MPEG-4 Part 10 Tutorials
[17]MPEG-4: ISO/IEC JTCI/SC29/WG11, "ISO/IEC 14 496:2000-2: "Information on Technology-Coding of Audio-Visual Objects"-Part 2: Visual," ISO/IEC, 2000.
.