MISB MOTION IMAGERY STANDARDS BOARD	MISB TRM 1404.1
TECHNICAL REFERENCE MATERIAL	
H.264 Compression Principles	**25 June 2015**

1 Scope

This TRM outlines the core principles in applying H.264 compression. Adherence to a common framework and understanding affords the greatest flexibility in choosing compression profiles and levels that meet desired capabilities.

2 References

2.1 Normative References

[1] ITU-T Rec. H.264 (04/2013), Advanced Video Coding for Generic Audiovisual Services
[2] MISP-2015.3 Motion Imagery Standards Profile, Jun 2015

3 Revision History

Revision	Date	Summary of Changes
1404.1	06/25/2015	Changed luminance to Luma consistent with SMPTE usage (Luma = gamma-corrected signals)

4 H.264 Compression: Understanding Profiles and Levels

ITU-T Recommendation H.264 (06/2011) [1] is a complex video coding standard that addresses a wide range of applications. It is impractical for most systems to fully implement all algorithm options and the full bit-stream syntax. The H.264 algorithm contains many options that may be used in the generation of a compressed video bit-stream, such as allowed video Chroma formats, sample bit-depths, slice and frame structures, motion compensation and prediction models, entropy coding, and transform and quantization choices. Such options impact the complexity of processing required to encode or decode a video sequence; the more complex this processing, the greater the number of calculations required per pixel.

An equally important system consideration is the volume of data processed. High pixel densities and high frame rates lead to a large number of pixels processed per second. More calculations per each second are needed as the number of pixels increases. Large frame sizes and high frame

rates also increase the amount of encoder/decoder memory needed for both the encoded bit-stream and the decoded video sequence.

ITU-T Recommendation H.264 defines *profiles* and *levels* that serve as points of common interoperability between conformant H.264 implementations. Profiles are subsets of the bit-stream syntax of H.264 that limit the algorithm options that may be used when encoding video. Within the boundaries established by a profile there may still be wide variation in the computing resources required by encoders and decoders as frame size and frame rate changes. Levels are then used to place constraints on the memory and processing throughput required during encoding and decoding a bit-stream.

Encoders claiming conformance to a particular profile and level must generate bit-streams conforming to that profile and level. They need not exercise all algorithm options in a particular profile, but they are constrained to use only those options allowed by the profile. Decoders, on the other hand, claiming conformance to a particular profile and level are expected to *fully* support that profile and level. Thus, a decoder may not pick and choose algorithm options within a profile; and, it is required to fully support the memory and processing throughput constraints within a level. Decoder compliance to a given profile and level allows system designers and users of a particular decoder to determine the types of H.264 bit-streams that may be reliably decoded.

Profiles group algorithm options into commonly used sets that are geared toward different application spaces. Algorithm complexity does not strictly increase with higher profiles, since some encoding options that are absent in higher profiles are present in lower profiles (see feature chart). Decoders compliant with higher profiles are required to decode bit-streams, or a constrained subset of bit-streams, compliant with lower profiles. For example, a decoder compliant with the Progressive High profile is not required to decode interlaced bit-streams from lower profiles; however, it must decode non-interlaced ones.

These parameters indicate that intra frame prediction only is allowed and buffering of the decoded pictures is not necessary since the Intra profiles only allow IDR pictures.

Table 1, derived from H.264/Annex A [4] of the standard and inspired by the Wikipedia page (https://en.wikipedia.org/wiki/H.264/MPEG-4_AVC) (somewhat out of date), lists the H.264 profiles and the algorithm options allowed for each profile.

The set of profiles listed in the table is current as of the [1] edition of the H.264 standard. Table 1 is a useful tool to gauge if an algorithm option is available in a particular profile; however, H.264/Annex A [1] should be consulted for specific details, and to determine if there are any level-related constraints placed on any of the profiles.

Table 1: H.264 Profiles

Option \ Profile	Baseline	Constrained Baseline	Main	Extended	High	Progressive High	High 10	High 4:2:2	High 4:4:4 Predictive	High 10 Intra	High 4:2:2 Intra	High 4:4:4 Intra	CAVLC 4:4:4 Intra
Chroma formats	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0 4:0:0	4:2:0 4:0:0	4:2:0 4:0:0	4:2:0 4:2:2 4:0:0	4:2:0 4:2:2 4:4:4 4:0:0	4:2:0 4:0:0	4:2:0 4:2:2 4:0:0	4:2:0 4:2:2 4:4:4 4:0:0	4:2:0 4:2:2 4:4:4 4:0:0
Sample bit depth	8	8	8	8	8	8	8 to 10	8 to 10	8 to 14	8 to 10	8 to 10	8 to 14	8 to 14
FMO	Yes	No	No	Yes	No	No	No	No	No	No	No	No	No
ASO	Yes	No	No	Yes	No	No	No	No	No	No	No	No	No
Redundant slice data	Yes	No	No	Yes	No	No	No	No	No	No	No	No	No
Data partitioning	No	No	No	Yes	No	No	No	No	No	No	No	No	No
SI and SP slices	No	No	No	Yes	No	No	No	No	No	No	No	No	No
B slices	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No
Interlaced coding	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
CABAC	No	No	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
8x8 vs. 4x4 transform adaptation	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Transform & de-quantization Bypass (lossless coding mode)	No	No	No	No	No	No	No	No	Yes	No	No	Yes	Yes
Quantization scaling matrices	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Separate C'_b and C'_r QP control	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Monochrome (4:0:0)	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Separate color plane coding	No	No	No	No	No	No	No	No	Yes	No	No	Yes	Yes
IDR pictures only	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes
Deblocking filter	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Not Required	Not Required	Not Required	Not Required
Intra Constraint Set[1]	No	No	No	No	No	No	No	No	No	Yes	Yes	Yes	Yes

[1] The Intra coding profiles have a common set of requirements. These include the following constraints on sequence parameter set variables:

1) *max_num_ref_frames* = 0
2) When *vui_parameters_present_flag* = 1 and *bit-stream_restriction_flag* = 1
    a. *max_num_reorder_frames* = 0
    b. *max_dec_frame_buffering* = 0

H.264 levels place limits on required encoder/decoder memory and processing throughput. Higher levels require more memory (larger Coded Picture Buffer (CPB) and Decoded Picture Buffer (DPB)) and greater processing throughput (MBs/sec processed) in order to accommodate larger frame sizes and/or higher frame rates. H.264/Annex A [1] provides some profile-specific constraints tied to encoder/decoder levels.

These constraints address memory and processing throughput issues as well. Table 3 gives the maximum throughput (*MaxMBPS* in macroblocks/sec), maximum frame size (*MaxFS* in macroblocks/frame), decoded picture buffer (DPB) size (*MaxDpbMbs* in macroblocks), coded picture buffer (CPB) size (*MaxCPB* in bits) and maximum video bitrate (*MaxBR* in bits/sec). Note that the *MaxCPB* size and *MaxBR* are a function of layer (VCL-Video Coding Level vs. NAL-Network Abstraction Layer).

The H.264 standard introduces two variables, *cpbBrVclFactor* (bits/sec), and *cpbBrNalFactor* (bits/sec), that are used as normalization factors for its level limits table (H.264/Annex A, Table A-1). These values are then increased for the higher profiles, which are expected to be used with larger frame sizes and higher frame rates, and therefore will generate higher bitstream bitrates. Table 2 provides the normalization factors across various profiles.

Table 2: Normalization Factors for Various Profiles

	Baseline, Constrained Baseline, Main, Extended	**High, Progressive High**	**High 10, High 10 Intra**	**High 4:2:2, High 4:2:2 Intra, High 4:4:4 Predictive, High 4:4:4 Intra, CAVLC 4:4:4 Intra**
cpbBrVclFactor	1000 bits/s	1250 bits/s	3000 bits/s	4000 bits/s
cpbBrNalFactor	1200 bits/s	1500 bits/s	3600 bits/s	4800 bits/s

Simple multiplicative factors of 1.25, 3 and 4 have been used to scale these variables for the higher profiles. Table 3 incorporates these scale factors into the values given for the *MaxCPB* size and *MaxBR* so that the normalization factors for VCL vs. NAL do not change as a function of profile.

Table 3 does not fully reproduce Table A-1 from the standard [1]; it only contains the limits discussed above. Table A-1 gives additional level limits. There is a minimum compression ratio, *MinCR*, maximum vertical motion vector range, *MaxVmvR*, and maximum number of motion vectors per two consecutive macroblocks, *MaxMvsPer2Mb*. These limits do not remove algorithm options, but rather constrain the level of processing. Other tables within H.264/Annex A contain some of the profile-specific level limits that serve to limit encoder and decoder processing. Interested readers should consult H.264/Annex A.

Table 3: H.264 Levels

				Baseline, Extended, Main Profiles		High Profile		High 10, High 10 Intra		High 4:2:2, High 4:2:2 Intra		High 4:4:4 Pred, High 4:4:4 Intra, CAVLC 4:4:4 Intra	
Level	Maximum MB/Sec	Max Frame Size (MBs)	MaxDPB (MBs)	MaxCPB VCL (1000 bits) NAL (1200 bits)	Maximum Bitrate VCL (1000 bits/sec) NAL (1200 bits/sec)	MaxCPB VCL (1000 bits) NAL (1200 bits)	Maximum Bitrate VCL (1000 bits/sec) NAL (1200 bits/sec)	MaxCPB VCL (1000 bits) NAL (1200 bits)	Maximum Bitrate VCL (1000 bits/sec) NAL (1200 bits/sec)	MaxCPB VCL (1000 bits) NAL (1200 bits)	Maximum Bitrate VCL (1000 bits/sec) NAL (1200 bits/sec)	MaxCPB VCL (1000 bits) NAL (1200 bits)	Maximum Bitrate VCL (1000 bits/sec) NAL (1200 bits/sec)
1	1485	99	396	175	64	218.75	80	525	192	700	256	700	256
1b	1485	99	396	350	128	437.5	160	1050	384	1400	512	1400	512
1.1	3000	396	900	500	192	625	240	1500	576	2000	768	2000	768
1.2	6000	396	2376	1000	384	1250	480	3000	1152	4000	1536	4000	1536
1.3	11880	396	2376	2000	768	2500	960	6000	2304	8000	3072	8000	3072
2	11880	396	2376	2000	2000	2500	2500	6000	6000	8000	8000	8000	8000
2.1	19800	792	4752	4000	4000	5000	5000	12000	12000	16000	16000	16000	16000
2.2	20250	1620	8100	4000	4000	5000	5000	12000	12000	16000	16000	16000	16000
3	40500	1620	8100	10000	10000	12500	12500	30000	30000	40000	40000	40000	40000
3.1	108000	3600	18000	14000	14000	17500	17500	42000	42000	56000	56000	56000	56000
3.2	216000	5120	20480	20000	20000	25000	25000	60000	60000	80000	80000	80000	80000
4	245760	8192	32768	25000	20000	31250	25000	75000	60000	100000	80000	100000	80000
4.1	245760	8192	32768	62500	50000	78125	62500	187500	150000	250000	200000	250000	200000
4.2	522240	8704	34816	62500	50000	78125	62500	187500	150000	250000	200000	250000	200000
5	589824	22080	110400	135000	135000	168750	168750	405000	405000	540000	540000	540000	540000
5.1	983040	36864	184320	240000	240000	300000	300000	720000	720000	960000	960000	960000	960000
5.2	2073600	36864	184320	240000	240000	300000	300000	720000	720000	960000	960000	960000	960000

It is interesting to note that H.264 levels do not explicitly state specific frame sizes or frame rates that must be supported. Rather limits are given for a *maximum* frame size specified in macroblocks/frame and the number of macroblocks/sec that must be processed. While Table A-1 does contain minimum compression ratios (*MinCR*), they are very low (largest value of *MinCR* is 4). Rather, the *MaxBR* and *MaxCPB* limits control the compression ratio in most systems. The reason that the H.264 limits are expressed in this fashion is that at its core, the H.264 compression system is a streaming macroblock processor. By specifying level limits in terms of macroblocks and macroblocks/sec, H.264 can accommodate a wide range of frame sizes, aspect ratios and frame rates. Encoders may then be tailored to a specific frame size, aspect ratio and frame rate. An encoder need only generate one compliant bitstream for a specific profile at a given level. Decoders must be able to decode all code-streams compliant to their supported profile and level including lower levels as well.

4.1 Macroblocks and Color Spaces

To better understand the level limits of H.264, we must understand macroblocks. The H.264 standard defines a macroblock as "A 16x16 block of Luma samples and two corresponding blocks of Chroma samples of a picture that has three sample arrays, or a 16x16 block of samples of a monochrome picture or a picture that is coded using three separate color planes." For most video coding applications the input data is in a Luma (corresponding to brightness) and Chroma (corresponding to color) space. Human vision is very sensitive to changes in light intensity (Luma), but less sensitive to changes in color (expressed as Chroma in this case), so it makes sense to handle these components differently.

Because the human visual system is tolerant to Chroma subsampling, a majority of video compression schemes make use of this technique. One of the most common Luma/Chroma sampling formats is 4:2:0, which corresponds to the 2x2 Chroma subsampling. A 4:2:0 sampling reduces Chroma resolution by one half in both the horizontal and vertical dimensions. This reduces the Chroma density by one-quarter its original density. The total quantity of data across all three components (Luma and two Chroma) is thereby reduced by a factor of two.

Tying this back to the definition of a macroblock we conclude a macroblock for a 4:2:0 source consists of a 16x16 block of Luma samples (Y'), an 8x8 block of C'_b Chroma samples, and an 8x8 block of C'_r Chroma samples. This arrangement is shown in Figure 1.

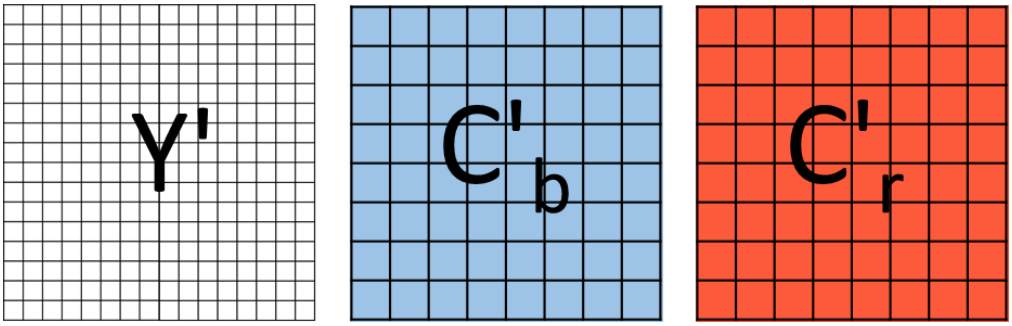


Figure 1. Macroblock Structure for 4:2:0 Luma and Chroma

4.2 Macroblocks, Frame Sizes and Aspect Ratio

Table 3 specifies a maximum frame size ($MaxFS$) in macroblocks, but it says nothing regarding the aspect ratio (width/height) of the frame size. In the following discussion we focus on the Luma component knowing that similar arguments hold for the Chroma components. Components must be composed of an integral number of macroblocks in width and height. For the Luma component this means that the width and height of a frame is a multiple of 16. If a video frame is not an integer multiple of 16 then the frame is either cropped down to the nearest multiple of 16 or padded out to the nearest multiple of 16. This choice is implementation dependent, but some standards bodies (SMPTE, DVB, BDA, etc.) have defined how various frame sizes map into H.264.

Annex A of the H.264 standard does place constraints on the aspect ratio allowed. Having frames that are 1000 macroblocks wide and one macroblock high (or vice versa) is not particularly useful. The constraints from Annex A are:

$$PicWidthInMbs \leq \sqrt{MaxFS \times 8} \qquad \text{Equation 1}$$

$$FrameHeightInMbs \leq \sqrt{MaxFS \times 8} \qquad \text{Equation 2}$$

$$PicWidthInMbs \times FrameHeightInMbs \leq MaxFS \qquad \text{Equation 3}$$

The constraints are very loose. They allow too wide a range of aspect ratios and do nothing to prevent the "many macroblocks wide by one macroblock high" problem. For example, let's assume that our decoder supports level 4 where $MaxFS = 8192$ macroblocks. The above equations reduce to:

$$PicWidthInMbs \leq 256$$
$$FrameHeightInMbs \leq 256$$
$$PicWidthInMbs \times FrameHeightInMbs \leq 8192$$

These conditions are easily met and do not prevent a frame size for *PicWidthInMbs* = 256 and *FrameHeightInMbs* = 1, leading to an aspect ratio of 256:1. The problem lies in using *MaxFS* in the first two equations. Nothing ties the true frame size into the equations, which allows for greatly distorted aspect ratios. The following formulation results in a much more useful result:

Let $FS = PicWidthInMbs \times FrameHeightInMbs$

Where, $FS \leq MaxFS$ Equation 4

and, $PicWidthInMbs \leq \sqrt{FS \times AR_{limit}}$
$FrameHeightInMbs \leq \sqrt{FS \times AR_{limit}}$

FS is the desired picture size in macroblocks and AR_{limit} is the preferred aspect ratio limit. Given the above definitions the following is derived:

$$PicWidthInMbs \leq \sqrt{FS \times AR_{limit}}$$

$$FrameHeightInMbs \geq \frac{FS}{\sqrt{FS \times AR_{limit}}} = \frac{1}{\sqrt{AR_{limit}}}\sqrt{FS}$$

$$AR = \frac{PicWidthInMbs}{FrameHeightInMbs} \leq \frac{\sqrt{FS \times AR_{limit}}}{\frac{\sqrt{FS}}{\sqrt{AR_{limit}}}} = AR_{limit}$$

So the aspect ratio, AR, will be less than or equal to AR_{limit} and will be exactly equal if the inequalities are all met with equality. The above equation represents an upper bound on AR; the lower bound is simply $1/AR_{limit}$ obtained by interchanging width for height. Combining these results yield:

$$FS = PicWidthInMbs \times FrameHeightInMbs$$

$$FS \leq MaxFS$$

$$\sqrt{FS/AR_{limit}} \leq PicWidthInMbs \leq \sqrt{FS \times AR_{limit}} \qquad \text{Equation 5}$$

$$\sqrt{FS/AR_{limit}} \leq FrameHeightInMbs \leq \sqrt{FS \times AR_{limit}}$$

$$\frac{PicWidthInMbs}{FrameHeightInMbs} = AR \in \left[1/AR_{limit}, AR_{limit}\right]$$

Using these expressions the constraints of any H.264 level comply and the range of allowed aspect ratios is managed. For MISP-compliant code-streams an aspect ratio limit of $AR_{limit} = 4$ is recommended. This should be more than enough for standard applications. Abel Gance's Napoleon (1927) was shot as three 4:3 frames shown simultaneously to create a 4:1 aspect ratio, called Polyvision. Napoleon (1927) was the only movie released in this format and it is the largest known aspect ratio for a commercial film. Disney's Circle Vision 360° uses nine 4:3 cameras to create a 12:1 aspect ratio projection. Circle Vision 360° is used solely in the Disney theme parks. Any 360° projection should be making use of the Multiview Video Coding (MVC) options in Annex H of H.264.

From a decoder perspective how is this information best used? A decoder's H.264 level limit places an upper bound on allowed frame size measured in macroblocks, $MaxFS$. Enforcing the aspect ratio limit, AR_{limit}, eliminates the "wide and short" or "tall and skinny" frame sizes.

All MISP-conformant encoders should produce bitstreams with aspect ratios in the range of $\left[1/_{AR_{limit}}, AR_{limit}\right]$. All MISP-conformant decoders shall support all combinations of $PicWidthInMbs$ and $FrameHeightInMbs$ that satisfy the aspect ratio limit constraint and number of macroblocks/frame constraint as given in equation set 5 consistent with their H.264 level as described in Annex A [1].

4.2.1 Informative Example 1

The information presented in this section is for informative purposes only. It may be useful for the reader to work through some of the example to obtain a better understanding and familiarity with the concepts discussed.

Suppose an encoder supports a nominal frame size of 100 macroblocks per frame and enforces an aspect ratio limit of four. At the extreme ends of possible frame sizes a frame that is 20 macroblocks wide and 5 macroblocks high or 5 macroblocks wide and 20 macroblocks high can be created. There are many other frame sizes between these extremes that require "approximately" 100 macroblocks per frame. Let one hundred macroblocks be the "nominal" frame size, and let the total number vary. Less than 100 macroblocks per frame is allowed, but not to exceed 100 as this is an encoder processing limit. For any given frame size use as many macroblocks as possible to maximize the frame size. Replacing FS with $FS_{nominal}$ in equation 5 and plugging in the parameters gives:

$$PicWidthInMbs \times FrameHeightInMbs \leq 100$$

$$5 \leq PicWidthInMbs \leq 20$$

$$5 \leq FrameHeightInMbs \leq 20$$

$$AR \in [0.25, 4]$$

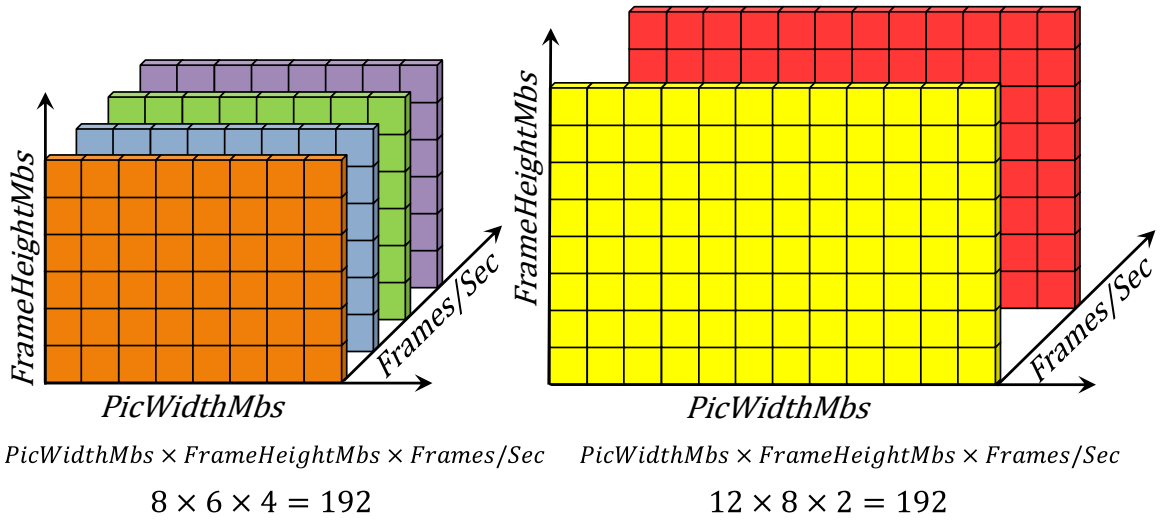
Table 4 lists all possible frame sizes subject to the above constraints. The widths and heights in macroblocks for all frames are given along with the true aspect ratio of each frame and actual frame size in macroblocks. As expected, not all $PicWidthInMbs, FrameHeightInMbs$ combinations yield a frame size of exactly 100 macroblocks, but the aspect ratio bounds are strictly obeyed.

Table 4: Frame Size Example

$FS_{nominal}$: 100		AR_{limit}: 4	
PicWidthMbs	**FrameHeightMbs**	**Actual AR**	**FS_{actual}**
5	20	0.250	100
5	19	0.263	95
5	18	0.278	90
5	17	0.294	85
6	16	0.375	96
6	15	0.400	90
7	14	0.500	98
7	13	0.538	91
8	12	0.667	96
9	11	0.818	99
10	10	1.000	100
11	9	1.222	99
12	8	1.500	96
13	7	1.857	91
14	7	2.000	98
15	6	2.500	90
16	6	2.667	96
17	5	3.400	85
18	5	3.600	90
19	5	3.800	95
20	5	4.000	100

4.3 Macroblocks and Frame Rates

Table 3 gives level limits on the maximum number of macroblocks/sec that a decoder is required to process as $MaxMBPS$. Together, $MaxMBPS$ and $MaxFS$ determine the frame sizes and frame rates that a decoder must support. Specifying level limits in this fashion allows for flexibility in making frame size versus frame rate tradeoffs. Implementations may spend the macroblocks as they see fit without being locked into specific frame sizes and frame rates. Figure 2 illustrates two scenarios with different frame rates and different frame sizes, but an equal number of macroblocks/sec. Any profile appropriate, level 1 compliant decoder will be capable of decoding both of these streams.



$PicWidthMbs \times FrameHeightMbs \times Frames/Sec$ $PicWidthMbs \times FrameHeightMbs \times Frames/Sec$

$8 \times 6 \times 4 = 192$ $12 \times 8 \times 2 = 192$

Figure 2: Different Frame Rates, Different Frame Sizes, Same Number of Macroblocks/sec

4.3.1 Informative Example 2

In this example, given the frame rate the number of macroblocks in a frame and the number of macroblocks/sec the system frame rate is computed. From this information the proper H.264 level limits that apply can be determined. First, a review of the calculations to determine the number of macroblocks in a H.264 compressed frame indicates that each frame be an integral number of macroblocks wide and high. Therefore, the frame dimensions in pixels $(FrameWidth, FrameHeight)$ must be a multiple of 16. Assume in these examples that any partial macroblocks are "rounded-up". In other words, the image is padded rather than cropped to meet the multiple-of-16 requirement. The ceiling function, indicated as $\lceil \cdot \rceil$ in the equations below, performs this function.

$$PicWidthInMbs = \left\lceil \frac{FrameWidth}{16} \right\rceil$$

$$FrameHeightInMbs = \left\lceil \frac{FrameHeight}{16} \right\rceil$$

$$FrameMBPS = PicWidthInMbs \times FrameHeightInMbs \times FrameRate$$

$$= FS \times FrameRate$$

Note: H.264 has a maximum frame rate of 172 Hz.

Table 5 gives examples of different frame sizes and frame rates and their corresponding sizes in terms of macroblocks/frame and macroblocks/sec. It also lists the corresponding H.264 level for each case. When determining the appropriate level, the frame size in macroblocks and the number of macroblocks/sec must be checked against Table A-1 [1]. The level limits in Table A-1 cannot be exceeded, so the level slightly larger than the compute frame size or macroblock rate is chosen. It is possible to come up with different answers, since one level limit constraint may be met before the other. In this case the higher level of the two is chosen.

The highlighted cells in Table 5 indicate which constraint – the number of MBs per frame or the number of MBs per second – determine the H.264 system Level. For example, 720x480 30 Hz indicates H.264 Level 2.2 based on the number of macroblocks/frame, but Level 3 based on the number of macroblocks/sec. Thus, Level 3, the greater of the two is selected. If its frame rate

were reduced to 15 Hz, resulting in 20250 MBs/sec, then it could be Level 2.2. A system common to Infrared might be 512x512 15 Hz; in this case, the number of macroblocks per frame dictates that Level 2.2 is required. A High Definition (1920x1080 30 Hz) system requires H.264 Level 4, while the Wide Area Sensor (WAS) (2048x2048 2 Hz) system requires Level 5. Wide Area Sensors typically have large frame sizes with low frame rates, so the frame size criteria generally will determine the required Level.

Table 5: Macroblocks/sec Examples

Frame Size (pixels)		Frame Size (MBs)		Frame Rate	Frame Size Criteria		MBs/s Criteria		Selected
Width	Height	Width	Height	Hz	MBs/frame	Level	MBs/s	Level	Level
720	480	45	30	30	1350	2.2	40500	3	3
512	512	32	32	15	1024	2.2	15360	2.1	2.2
1920	1080	120	68	30	8160	2.2	244800	4	4
2048	2048	128	128	2	16384	5	32768	3	5

4.4 Decoder Conformance and Points of Interoperability

The H.264 standard states, "A decoder claiming conformance to a specific profile and level shall be able to decode successfully all conforming bit-streams specified for decoder conformance in sub clause C.3, provided that all sequence parameter sets and picture parameter sets referred to in the VCL NAL units, and appropriate buffering period and picture timing SEI messages are conveyed to the decoder, in a timely manner, either in the bit-stream (by non-VCL NAL units), or by external means not specified by this Recommendation | International Standard." MISP-conformant decoders claiming conformance to a given H.264 profile and level should fully support that profile and level.

It is not permissible for a decoder to claim conformance to a profile or level and only partially support the range of features, frame sizes, macroblock rates, etc. within that profile and level. A *MISP-complaint encoder* conforming to a given H.264 profile and level need only produce one bitstream conforming to that profile and level. Conformance requirements for decoders encompass a broader class of bitstreams than those for encoders. A conforming decoder must be able to decode *all* bitstreams conforming to their profile and level. This is certainly difficult to test and a high standard to meet.

To improve system interoperability and testing of decoders, a set of "Points of Interoperability" have been created (see MISP [2]) recommended for current implementations, and for legacy systems based on earlier versions of the MISP (pre dating MISP 2015.1). These points of interoperability define a wide range of frame sizes and frame rates that span most of the H.264 levels. MISP-conformant decoders will be able to decode bitstreams conforming to those points of interoperability within their particular H.264 level. Systems without a MISP-conformant decoder will hopefully be able to support one or more of the points of interoperability, thus enabling some measure of interoperation with MISP-conformant systems. MISP-conformant encoders should try to support the points of interoperability within their own conformance level.

For the widest possible interoperation between systems, current MISP decoders may wish to consider adding support for MPEG-2 decoding as suggested in the Legacy table.