

NAL-SIM v1.0 User Manual

Written by Hadi Hadizadeh

Multimedia Communications Lab, Simon Fraser University

All rights reserved, May 2009.

Email: hha54@sfu.ca

1 Introduction

As shown in Figure 1, the simulator has five main parts. Part A is the encoder box, Part B is the Channel box, Part C is the decoder box, Part D is an H.264/AVC stream analyzer, and Part E is the output window. The description of each part is explained in the following sections.

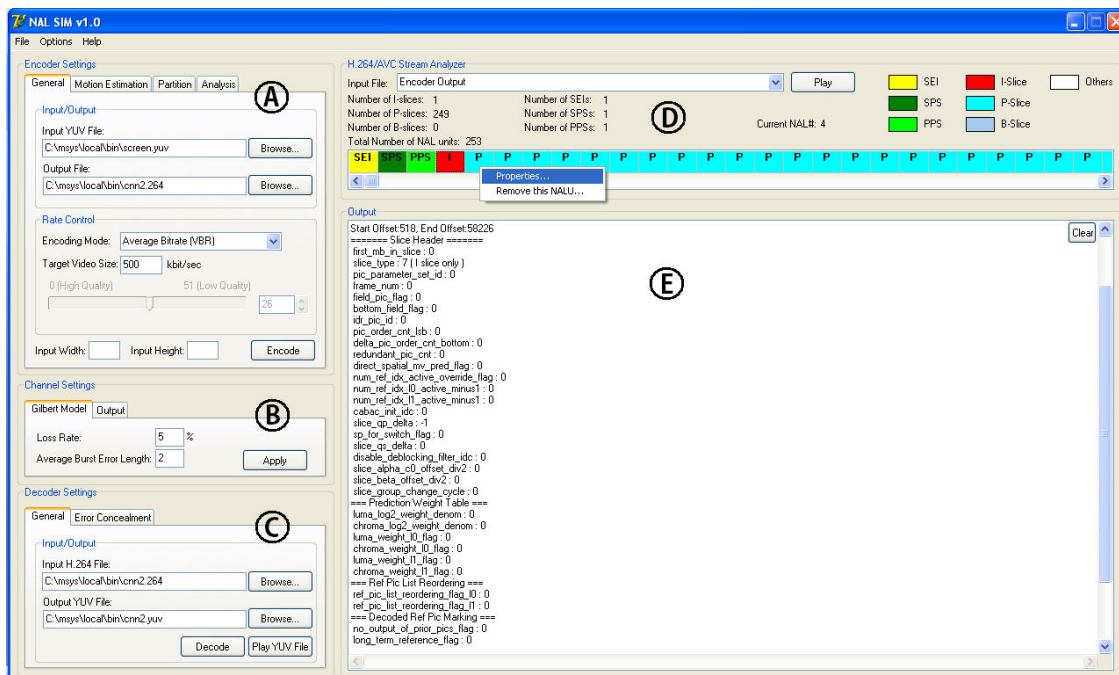


Figure 1: A snapshot of the designed simulator

2 Part A: Encoder Settings

In this section, the description of all available options in the encoder box is explained.

2.1 General Tab

This tab of the encoder box has shown in Figure 2. The description of each option in this tab is explained in the following subsections.

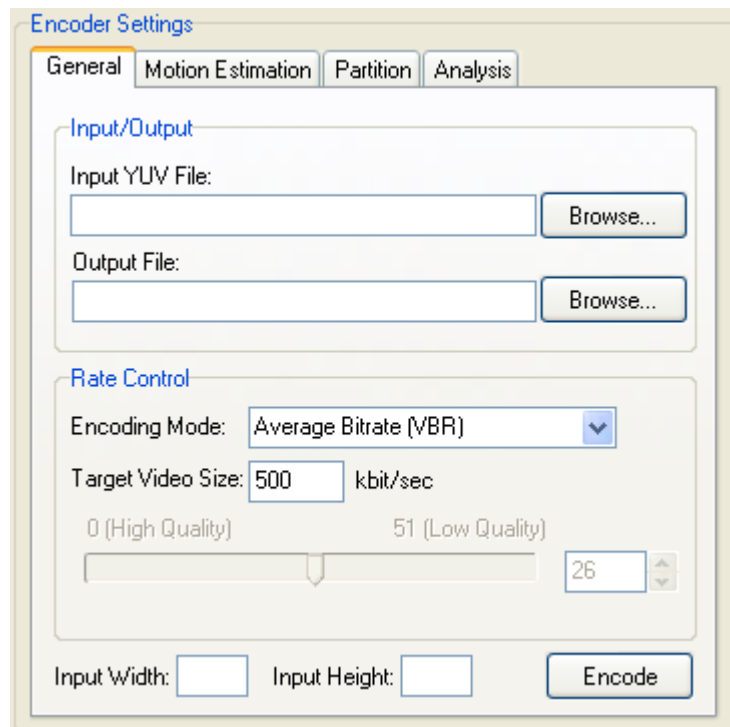


Figure 2: "General" Tab of the encoder box

2.1.1 Input/Output

In this section, you can specify the input YUV file that must be encoded and also an output H.264 file. The output file is always overwritten. Note that the extension of the input file (.yuv) or the output file (.264) must always be entered

2.1.2 Rate Control

In this section, the following three rate control methods can be chosen:

1. **Average Bitrate (VBR):** This mode will encode the input video at an *average* bitrate with only one single pass. In contrast to *CRF* mode (and *QP* mode) the resulting average bitrate is known in advance. Therefore it is easy to predict the final file (bitstream) size. A *higher* bitrate will result in a better visual quality, but of course it will also result in a bigger file (bitstream). A *lower* bitrate will result in a smaller file (bitstream), but it will also result in a worse visual quality.
2. **Constant Quantizer (QP Mode):** This mode is also known as the “QP Mode”. It will encode the input video using a *constant quantizer*, so you will choose the target *quantizer scale*, not the target *bitrate*. The quantizer is a measure for the amount of data loss: a *higher* quantizer means that more data will be lost, which results in a better compression (smaller file or bitstream), but also delivers worse visual quality. In contrast, a *lower* quantizer means that less data will be lost, which results in a better visual quality, but also compresses worse (larger file). H.264/AVC uses a quantizer scale between 0 and 51. The default quantizer value is 26. If you are targeting for a certain *level of quality* and do not care much about the final file (bitstream) size, then you might consider using the *QP* mode. But if you are targeting for a certain file size (or a certain average bitrate), then keep away from QP Mode. That is because the final size (the average bitrate) is completely *unpredictable* in this mode.
3. **Constant Rate Factor (CRF):** This mode is also known as the “CRF Mode” or “Constant Visual Quality” mode. It basically works similar to the QP Mode (see above), but it will encode with an *average* quantizer instead of a constant one. To be more precise, this mode encodes at a constant “rate factor”, which is derived from the specified quantizer. Internally CRF mode uses the same rate-control algorithm as x264's *ABR* mode, only *without* a target bitrate. The advantage of the *CRF* mode is that it suits the human perception much better than the *QP* mode. For example it will raise the quantizers in “fast” scenes where the loss will not be visible anyway and lower the quantizers in “slow” scenes. Therefore the *CRF* mode should give the same *subjective* quality as *QP* mode, but it usually achieves a significant higher compression. It is recommended to prefer *CRF* mode over *QP* mode, although *CRF* is a bit slower. When switching from *QP* to *CRF* mode, you may want to slightly lower the quantizer. This should give approximately the same file (bitstream) size as before, but better visual quality. Another important advantage of *CRF* mode is that it will benefit from *adaptive quantization (AQ)*, something that *QP* mode can not do. Please note that Even the *CRF* mode does not deliver “perfect” constant quality. A specific CRF value will only deliver *similar* quality for various sources, as long as you do not change any other settings. Using “slower” settings with the *same* CRF value will either produce a smaller file (bitstream) at same quality or a produce a file of the same size at a better quality. It is also possible that both, the size and the quality, will be increased. The “quality per size” ratio will be improved anyway.

Please note: The *CRF* mode should be preferred over the *QP* mode. Also the *Adaptive Quantization (AQ)* will be disabled in *QP* mode, while it is enabled (by default) in *CRF* mode. It must be pointed out that choosing the proper *quantizer* setting for a *CRF* (or *QP*) encode is not trivial. That is because visual quality is highly *subjective*: What some

people consider “good quality” other people will consider “horrible quality” - and vice versa. Furthermore the quantizer setting highly depends on contents of the video. Nevertheless a quantizer setting in the range between **16** and **32** should give satisfactory results in most cases. Using a quantizer *lower* than **16** usually is overkill, except for mastering purposes. Using a quantizer *higher* than **32** will result in almost *unwatchable* video. A quantizer of **22** seems to be reasonable for most purposes. Nevertheless material with few textures, like Anime and Cartoons, can cope with much *higher* quantizers. At the same time “real life” footage with a lot of textures might require much *lower* quantizers, especially in dark scenes. There also is a rule of thumb: Lowering the CRF value by 6 will double file size, lowering CRF by 1 will raise file size by ~12.5% (very roughly). Furthermore the common practice is as follows: Start with a low CRF value, such as *16*. Then raise the CRF value in steps of one, until the quality becomes intolerable. This way you will find the highest possible CRF value that still gives an accepted quality for your eyes. Once you found it, you can use that value for all your future encodes [53].

The width and height of the input YUV file should be specified using the lower two text boxes. Finally, to encode the input file, you must press button “Encode”. The output of the encoding process is written in the output window (Part E). In this output, the average PSNR value of the encoded video and also the speed of the encoding process (fps) are printed in the output.

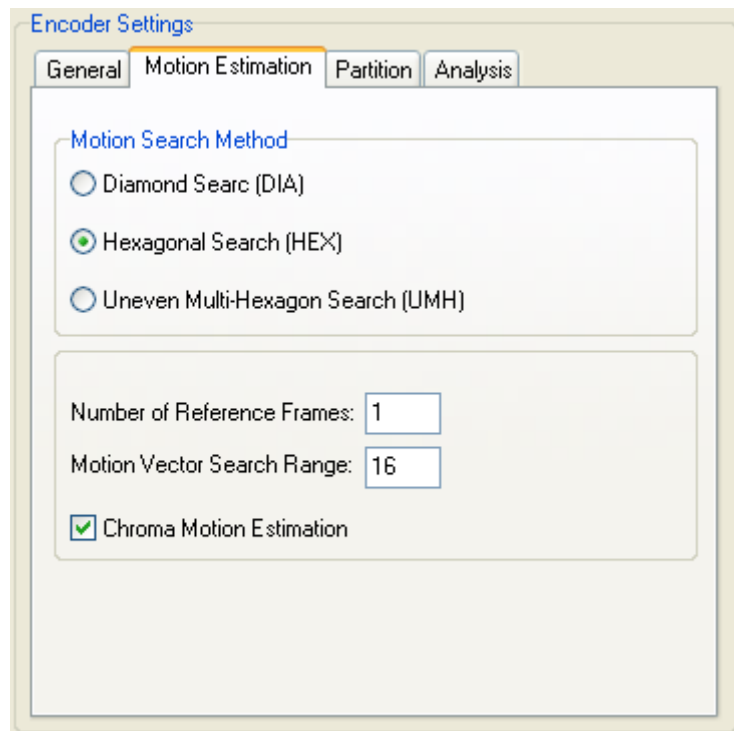


Figure 3: “Motion Estimation” tab of the encoder box

2.2 Motion Estimation Tab

This tab of the encoder box has shown in Figure 3. The description of each option in this tab is explained in the following subsections.

2.2.1 Motion Search Method

The following three motion search methods can be chosen:

- **Diamond Search (DIA):** Four sided shape analysis - This is the fastest method, but it also provides the worst quality. Use this for method only if encoding speed is more important than quality.
- **Hexagonal Search (HEX):** Six sided shape analysis – This is the *default* method. It provides reasonable quality and still works pretty fast.
- **Uneven Multi-Hexagon Search (UMH):** More detailed version of Hexagonal search - This method provides high quality, but works *slower* than the simple "HEX" method. If you prefer quality over speed, then use this method.

2.2.2 Motion Vector Search Range

This setting defines how many pixels are analyzed for motion estimation. Higher *range* values result in a more accurate analysis, but will also slow down the encoding speed significantly. Lower values will speed-up the encoding process, but will also result in a less accurate analysis. Note that high resolution material generally benefits more from higher *range* settings than low resolution material. That's because objects tend to move farther (with respect to pixels) in HD video. Anyway, the default value of **16** is sufficient for most videos! The "Diamond Search" and "Hexagonal Search" methods are even limited to maximum range of *16*. If quality is more important than encoding speed and if you are using the "Uneven Multi-Hexagon Search" method (or an even slower method), you may want to raise *range* to a value of **24** or even **32**. Depending on the selected ME method, the *range* value may be rounded up to a multiple of *two* or *four*.

2.2.3 Number of Reference frames

This setting controls how many frames can be referenced by P- and B-Frames. Higher values will usually result in a more efficient compression, which means better visual quality at same file size. Unfortunately more reference frames will require more time for encoding (and also a tiny bit more CPU power for playback). By default the number of reference frames is limited to **1**. Using more than **4** or **5** reference frames for "real life" footage should be avoided, as it won't improve the results any further. At the same time Anime and cartoons may benefit more from additional reference frames. Sometimes even the maximum of **16** reference frames may be helpful for such materials.

2.2.4 Chroma ME (Motion Estimation)

If this setting is enabled, then the color information (chroma) will be taken into account for motion detection, otherwise it will not. With "Chroma Motion Estimation" enabled the motion detection will be slower but more accurate. So it will usually produce a *higher* visual quality at the cost of some encoding time. Therefore it's recommended to always keep this setting **enabled**, except encoding speed is more important than visual quality.

2.3 Partition Tab

This tab of the encoder box has shown in Figure 4. The description of each option in this tab is explained in the following subsections.

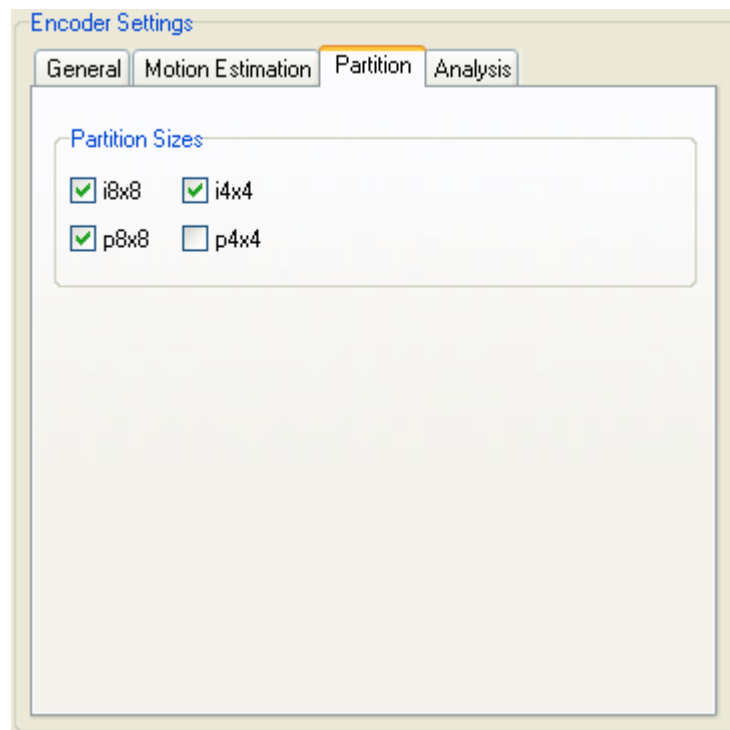


Figure 4: "Partition" tab of the encoder box

2.3.1 Partition Sizes

The following options can be chosen to determine the choice of macroblock partitions or macroblock sup-partitions for motion compensation process:

- **i8x8:** This setting enables the 8x8 partitions on I-Frames and thus improves the visual quality of these frames. It's recommended to keep this option enabled, if possible.
- **i4x4:** This setting enables the 4x4 partitions on I-Frames and thus improves the visual quality of these frames. It's recommended to keep this option enabled.
- **p8x8:** This setting enables the 8x8, 8x16, and 16x8 partitions on P-Frames and thus improves the visual quality of these frames. It's recommended to keep this option enabled.
- **p4x4:** This setting enables the 4x4, 4x8, and 8x4 partitions on P-Frames, but usually the quality improvement will be negligible. Therefore this option is *not* worth the additional encoding time and thus can safely be turned *off*, especially for high resolutions.

Remarks: The macroblocks can be subdivided into 16x8, 8x16, 8x8, 4x8, 8x4, and 4x4 partitions. Analyzing *more* of these partitions results in a more accurate prediction and thus improves the visual quality. Unfortunately this comes at the cost of additional encoding time. Generally it's recommended to keep *all* partition types *enabled*, except the for the "4x4 P-Frame" partitions. That's because the 4x4/4x8/8x4 partition search on P-Frames costs a significant amount of encoding time, but the gain in quality usually is negligible (only low resolution video *may* benefit). Note that some of the partition options depend on each other.

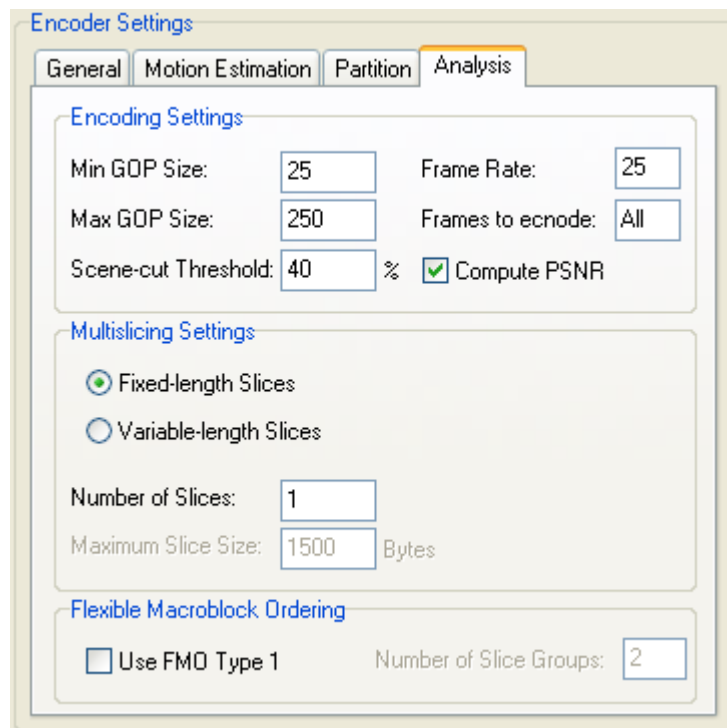


Figure 5: "Analysis" tab of the encoder box

2.4 Analysis Tab

This tab of the encoder box has shown in Figure 5. The description of each option in this tab is explained in the following subsections.

2.4.1 Encoding Settings

1. **Min GOP Size:** This setting controls the *minimum* number of frames between two IDR (Instantaneous Decoder Refresh) frames. IDR frames are similar to Key-frames in MPEG-4 ASP videos: Playback can only be started at an IDR frame, as no frame *after* the IDR frame will refer to a frame *before* the IDR frame. In H.264 this is *not* possible with "normal" I-Frames, because of the multiple references. So IDR frames are needed to allow seeking in the video. Nevertheless too many IDR frames would cause an inefficient encoding, so there's a *minimum* interval for IDR frames. As a rule of thumb, this value should equal the *framerate* of the video. For example a 25 fps video should use a value of **25**, a 29.97 fps video should use a value of **30** and so on.
2. **Max GOP Size:** In contrast to "Min IDR frame interval" this setting controls the *maximum* number of frames between two IDR frames. A *higher* value will result in a larger IDR frame interval and thus slowdown seeking; a *lower* value will result in a shorter IDR frame interval and thus improve seeking. As a rule of thumb, for encoding movies in error free environments, the IDR frame interval shouldn't be lower than the frame rate of the video multiplied with a factor of *10*. For example a 25 fps video should use at least a value of **250**, a 29.97 fps video should use at least a value of **300** and so on. Using even higher values will improve the compression at the cost of some seeking performance. Of course martial with many "long takes" and long "tracking shots" will benefit much more from long GOP's than martial which mainly consists of very short scenes. Please note that long GOP's will hurt error resilience, which may be a problem for steaming media.
3. **Scene Cut Threshold:** This setting controls the threshold for x264' *scene change* detection module. This way the encoder can put an I-Frame at every scene change (instead of an P- or B-Frame), which should lead to better looking scene cuts. A *lower* threshold results in a more aggressive scene change detection, which might be useful for very dark videos. In contrast a *higher* threshold will detect less scene changes. The default is **40** and should be suitable for most videos.
4. **Frame rate:** The frame rate of the video can be entered in this text box.
5. **Frames to encode:** This option determined the number of frames to be encoded. If all frames must be encoded, then write "**All**" in this text box.

6. **Compute PSNR:** This option disable/enable the computation of the PSNR value of the input video. To measure the actual encoding speed, disable this option.

2.4.2 Multislicing Settings

In this section, the following two options for using raster scan slices are provided:

- 1- **Fixed-length Slices:** Using this option, each frame is divided to a fixed number of ordinary raster scan slices of the same size (in macroblocks).
- 2- **Variable-length Slices:** Using this option, the maximum slice size (in bytes) can be set. This way, the resultant slices will have variable number of macroblocks. In fact, using this option, a new slice is opened and a certain number of encoded macroblocks are written into it until the maximum allowed slice size is reached. A value smaller than the MTU size can be used here to ensure that the outgoing slices fit well into each network's packet. For example, if the MTU size is 1500 bytes, then a value about 1400 bytes can safely be used.

2.4.3 Flexible Macroblock Ordering

To use FMO Type 1 (scattered slices or checkerboard pattern) check “FMO Type 1” and enter the number of desired slice groups in range 2-8. Enabling this option will disable Multislicing options and vice versa.

3 Part B: Channel Settings

In this section, the description of all available options in the channel box is explained.

3.1 Gilbert Model Tab

In this section, as shown in Figure 6, you can specify the loss rate and the average burst error length to determine the parameters of the Gilbert model described in Appendix I. The input to the channel is the output of the encoder. To drop NAL units of the input file, press “Apply”.

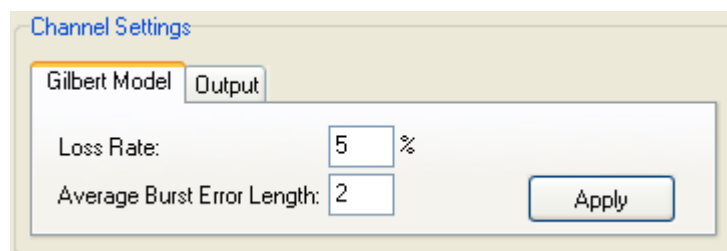


Figure 6: “Gilbert Model” tab of the channel box

3.2 Output Tab

You also need to specify an output file (see Figure 7) which is actually the input file which some of its NAL units have dropped by the Gilbert model. This output file can be analyzed or displayed in “H.264/AVC Stream Analyzer” section. Note that the extension of the output file (.264) must always be entered.

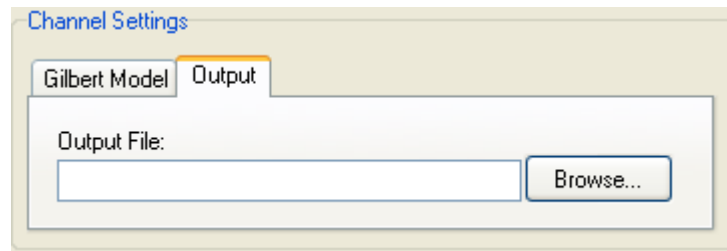


Figure 7: “Output” tab of the channel box

4 Part C: Decoder Settings

In this section, the description of all available options in the decoder box is explained.

4.1 General Tab

In this section, as shown in Figure 8, the input H.264 file to the decoder and also the output YUV file can be specified. The decoded YUV file can be displayed using an arbitrary YUV player. The address of the YUV player can be specified in menu “Options->YUV Sequence Player”. The default YUV player is “seqview.exe” which is a free software and is available in the main folder of the simulator. To decode the input video sequence press “Decode”. The output of the decoding process will be displayed in the “Output” window. Note that the extension of the input file (.264) or the output file (.yuv) must always be entered. Also, the input to the decoder can be any arbitrary H.264 file such as the output of the encoder or the output of the channel. The output file will always be overwritten.

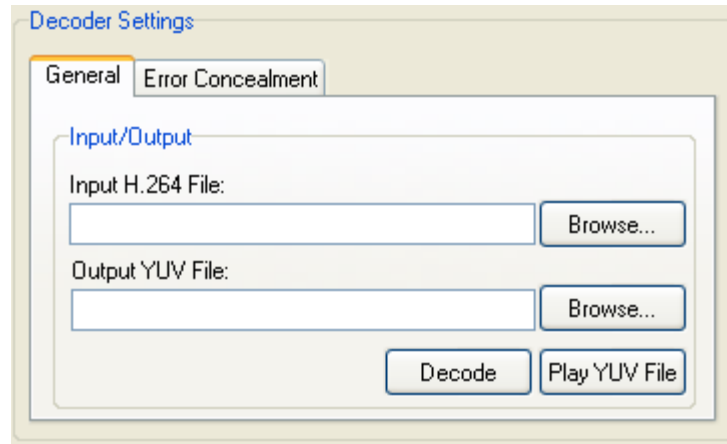


Figure 8: “General” tab of the decoder box

4.2 Error Concealment Tab

As shown in Figure 9, several error concealment methods for both I- and P-frames can be chosen in this section. The description of each of these methods can be found in Chapter 6.

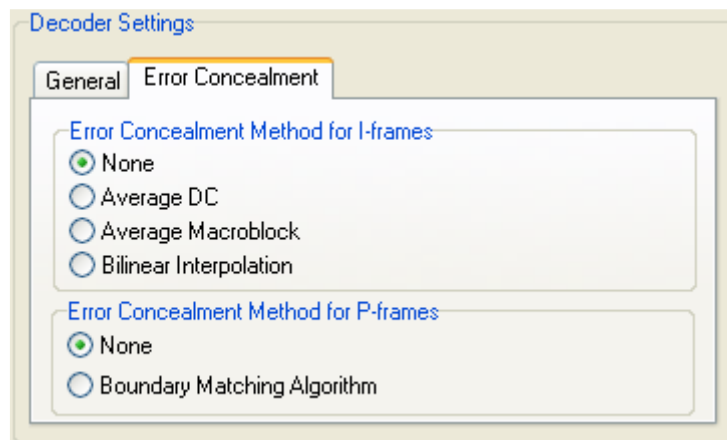


Figure 9: “Error Concealment” tab of the decoder box

5 Part D: H.264/AVC Stream Analyzer

In this section, the output of the encoder or the output of the channel can be visually analyzed or displayed. The input source can be chosen from the drop-down menu. To display the input source using the predefined H.264 video player press “Play”. The H.264

video player can be specified in menu “Options->H.264 Video Player”. The default player is VLC media player which is an open-source software and you can download it from [52].

As soon as you choose an input source, the content of it is analyzed and each NAL unit of the input source is displayed with a separate colored box. Each color represents one specific type of NAL units. For example, red boxes show NAL units which contain I-slices while light blue boxes show NAL units which contain P-slices. Also, the total number of each specific NAL unit is displayed in this window.

The stream analyzer allows the user to see the content of each NAL unit or to remove it. To do so, first select a NAL unit and then right click on it. To see the whole the header information of the selected NAL unit, select “Properties”. The header information will be displayed in the “Output” window. Also, to remove one NAL unit, select “Remove”. After removing one NAL unit from the selected input source, the modified source can be displayed (by pushing the “Play” button) or decoded using the decoder box.

Whenever you need to refresh the stream analyzer, you should select the input source from the drop-down menu once again even if the current source is the same as the previous one.

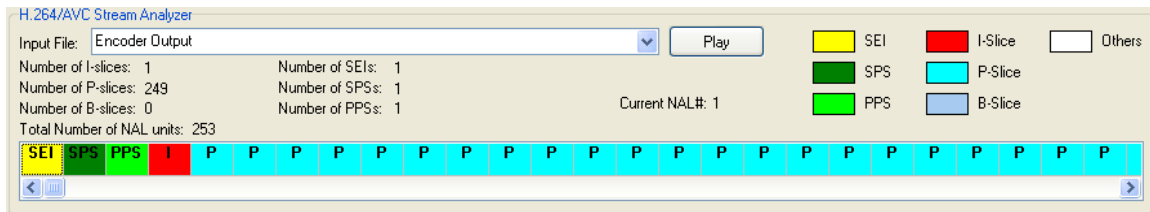


Figure 10: H.264/AVC stream analyzer box

6 Part E: Output Window

The output of each part is displayed in this section. To clear the output window, press the small button on the top right corner of the window.