SVIP Parameter File Syntax

1. Overview

The parameter file is interpreted on a line by line basis. Lines that begin with a "%" sign are comments and are ignored. Some parameters are "common" to both the encoder and the decoder. To properly decode a bit-stream, the same common parameters must be supplied to the decoder. Other parameters are either encoder or decoder specific and can be changed without re-encoding. An example parameter file is shown below. The meanings of the parameters are explained in Sections 2-4.

```
% COMMON PARAMETERS
WIDTH = 320
HEIGHT = 224
SPATIAL_LEVEL = 4
TEMPORAL_LEVEL = 2
SMALLEST_BLOCK = 2
N_COMPONENTS = 35
MAX_QLAYER = 15
NCELL = 150
FS_MODE = 0
RATES = 10+20+30+40
% DECODER SPECIFIC
INPUT = tcp:128.32.239.173/8800
OUTPUT = screen
DECODE_CELL = 38
DEPTH = 24
WRAP = 0
MAX_RATE = 20.0
% ENCODER SPECIFIC
ENC_INPUT = Raider12/frame.%04d
ENC_OUTPUT = bits.raider
```

2. Common Parameters

Parameter	Symbol	Default Value	Meaning
WIDTH	*w*	320	Picture width
HEIGHT	*h*	224	Picture height
SPATIAL_LEVEL	*s*	4	Number of levels of spatial subband decomposition employed
TEMPORAL_LEVEL	*t*	0	Number of levels of temporal subband (Haar) decomposition employed
SMALLEST_BLOCK	*a*	2	Refer to [1]. The block size use to code the DC-subband. Must be one of 1,2,4,8 and 16. With the use of *t* levels of temporal decomposition, then, from [1], the biggest block used is $a2^{t-1}$, which must also not exceed 16.
N_COMPONENTS	*m*	1	Refer to [1]. Each component becomes a transport packet. The DC subband is of size $wh2^{-2t}$, giving a total of $S = wh2^{-2t}a^{-2}$ spatial locations or blocks. To evenly distribute the *S* spatial locations into *m* components, *m* must divide *S*. Thus, *m* is any divisor of *S*, including 1 and *S*.
MAX_QLAYER	*q*	14	Short integers (16-bits) are used for internal representation of subband coefficients. Excluding the sign bit, this gives at most 15 bit-planes. This parameter determines the maximum number of bit planes that any subband can be coded. It cannot exceed 15, and generally should not be smaller than 12.
NCELL	*l*	200	Represents the number of layers, and thus the maximum rate of the encoded bit-stream.
FS_MODE	*f*	0	Whether frame-rate scalability mode is used. Valid values are 0 for not using frame-rate scalability mode, and 1 otherwise.
RATES	*R*	-	Used in frame-rate scalability mode. When *t* levels of temporal decompositions are used, there are *t*+1 available frame rates, namely, 1, ½, ¼, … , $1/2^t$ of the original rate. In that case, we need *t* number to represent the layer numbers in which the frame rates change. For instance, if 3

			levels of temporal decompositions are used, and RATES = A+B+C, then, at and below layer A, 1/8 of the original frame-rate will be obtained. Layers A+1,…,B corresponds to ¼of the original frame rate, and B+1,…,C corresponds to ½of the original frame rate. Layers above C achieve full frame-rate.

Notes on choosing parameters:

Due to the different characteristics, such as the amount of motion and details, of different input sequences, no one set of encoding parameter is optimal for all sequences. In this session, we discuss generally what effects some encoding parameters have on compression performance.

1. SPATIAL_LEVEL: Generally, the more levels of spatial decomposition yields higher compression performance at the cost of higher complexity. Generally, 4 or 5 levels should be used for image sizes around 320 by 224.
2. TEMPORAL_LEVEL: Ideally, compression performance should increase with the number of temporal decomposition used. This is true when we increase temporal decomposition in the range from 0 to 3. For more than 3 levels of temporal decomposition, little, no or negative gains are sometimes observed. Using more levels of temporal decomposition also entails higher computing costs.
3. SMALLEST_BLOCK: Generally speaking, sequences with great amount of details desire smaller block sizes and *vice versa*. Since typically several (3-4) levels of spatial decomposition may be used, the choice for this parameter is usually limited to 1, 2, or 4.
4. N_COMPONENTS: The bit-stream includes a 2 bytes header for every component per GOP. This should have minimal effect on compression performance except at low bit rates.

Layer Size Calculation:

The size of a layer L is dependent on the number of spatial locations $S = wh2^{-2s}a^{-2}$ by the following $L = 4S$ bytes per GOP.

3. Decoder Specific Parameters

PARAMTER	MEANING
INPUT	Normally signifies the name of the bit-stream. However, if it is of the form `tcp:`*`ipaddress/port`*, where *`ipaddress`* and *`port`* are the IP-address and port number to contact a video server, the decoder will contact the video server to start the streaming process.
OUTPUT	"screen" signifies screen output. Otherwise, frame by frame disk output is assumed. In the later case, `OUTPUT` represents the format argument of the output frames. For example, `Out/frame.%04d` will cause files frame.0000, frame.0001, and so forth to be generated in the directory "Out". See man-page for `printf`.
DEPTH	When decoding to screen only, and only for Unix players. The displayed image depth.
WRAP	When decoding from disk only. Whether to wrap around the bit-stream after reaching end of file. Valid values are 0 for terminate and 1 for wrap-around.
MAX_RATE	A parameter for smoothing the display of images. It is a floating point value, and the decoder will be artificially slowed to decode at a slower frame-rate than `MAX_RATE`. Thus, when getting streaming video of 12 fps and setting `MAX_RATE` to 12.0 may cause the decoder to crash since the decoder is now decoding slower than the source. Reasonable values are 1.5 to 2 times the actual desired frame-rate.
DECODE_CELL	When decoding from disk only. The number of layers to decode.

4. Encoder Specific Parameters

PARAMTER	MEANING
ENC_INPUT	Format string for the input frames. E.g., if input frames are in directory "Input", and are numbered as frame.001, frame.002 etc, then use `Input/frame.%03d.`
ENC_OUTPUT	File name of the output bit-stream.

References

[1] W. Tan and A. Zakhor, "Real-time Internet Video Using Error Resilient Scalable Compression and TCP-Friendly Transport Protocol", *IEEE Trans. Multimedia*, Vol. 1, No. 2, June 1999, pp 172-186.