Software manual for 3DV-ATM (reference software for MVC+D and 3D-AVC specifications)

Version: 3DV-ATM 13.1

Last update: September, 2014

# Summary:

This document contains a description of the usage and configuration of the 3DV-ATM (3D Video AVC based Test Model) software which serves as a reference software for MVC+D and 3D-AVC specifications of multiview video plus depth coding technologies. The software allows run-time configuration of the encoder which results in either MVC+D or 3D-AVC complied bitstreams production.

This document provides information how to build the software on Windows and Linux platforms. It contains a description of the usage and configuration for the binaries built from the software package.

# Building 3DV-ATM software

The 3DV-ATM source code is provided with Microsoft Visual Studio project files which allows building 3DV-ATM codec for 32 and 64 bits version of Microsoft Windows. Building of the 3DV-ATM for Windows platforms with use of other compilers is expected to be straightforward. Although the current version of the software was tested with Microsoft Visual Studio only.

To produce executable for linux platform, user can utilize provided make files to produce executables for 32 or 64 bits versions of linux. The current version of the software was tested with gcc 4.x compilers, however compilation of the source code with other compilers is expected to be straightforward.

As a result of build, stand-alone executable files /bin/lencod.exe and /bin/ldecod.exe implement encoder and decoder respectively.

1. Configuration files

The 3DV-ATM codec supports the multi-view plus depth (MVD) data coding and takes input parameters from configurations files. In the provided software package, example configuration files are provided under ./configs folder, which has the following contents.

Table 1 Main Texture Configurations

./configs:

mvc+d/ 3davc/ cam\_param/ decoder.cfg

./configs/mvc+d:

encoder\_texture.cfg texture\_additional.cfg depth\_additional.cfg encoder\_depth.cfg

./configs/3davc:

encoder\_texture.cfg texture\_additional.cfg depth\_additional.cfg encoder\_depth.cfg

./configs/cam\_param:

cam\_param\_kendo.txt cam\_param\_poznanhall2.txt cam\_param\_balloons.txt cam\_param\_gt\_fly.txt cam\_param\_lovebird1\_4\_33views.txt cam\_param\_newspaper.txt

… …

File ./configs/decoder.cfg is an example decoder configuration file for both MVC+D and 3D-AVC.

Under ./configs/mvc+d/ are example encoder configuration files for MVC+D (4 files).

Under ./configs/3davc/ are example encoder configuration files for 3D-AVC (4 files).

# Encoder run-time configuration parameters

In this section, the configuration parameters for a 3DV-ATM encoder are described.

Encoding configurations for texture and depth component are specified separately in two major input files, one for texture coding (see Section 1) and another for depth coding (see Section 2).

Note that each of the component (texture or depth) has an additional configuration file targeted for a view-specific parameters, such as view coding order, camera parameters and so on.

# Texture coding parameters

Table Main Texture Configurations

# Highest level controls

3DVCoding = 1 # 0: MVC-compatible (MVC+D); 1: AVC-compatible (3D-AVC)

NumberOfViews = 3 # Number of texture views to be coded

3DVCodingOrder = "T0D0D1D2T1T2" # Specifies coding order for MVD components views

3DVConfigFile = 3dv\_texture\_balloons.cfg # Sequence-specific configuration for current

# texture component of MVD data

# Normative switches for 3D-AVC only coding tool

#

DepthBasedMVP = 1 # Enable depth-based motion vector prediction for texture

SliceHeaderPred = 1 # Enable slice header prediction from texture to depth or

# depth to texture, depending on coding order.

PredSliceHeaderSrc = 2 # Parameter 1 for slice header prediction

PredRefListsSrc = 0 # Parameter 2 for slice header prediction

PredWeightTableSrc = 0 # Parameter 3 for slice header prediction

PredDecRefPicMarkingSrc = 2 # Parameter 4 for slice header prediction

VSP\_Enable = 1 # Enable VSP for dependent texture views

AaptiveLuminanceCompensation = 1 # Enable adaptive luminance compensation

AlcSearchRangeX = 20 # Parameter for ALC

AlcSearchRangeY = 6 # Parameter for ALC

RLESkip = 1 # Enable run-length coding for skip flag

# Encoder only switches: MVC+D and 3D-AVC

#

InterPredictionAtAnchorOff = 0 # Disable inter prediction at anchor picture(Default:1)

GradualViewRefresh = 0 # Enable Gradual View Refresh (0: disable/default, 1: enable)

# 3DV SEIs

#

SEI\_MultiviewAcquisitionInfoFlag = 0 # Enable to send Multiview acquisition SEI

SEI\_3DReferenceDisplaysInfo = 0 # Enable to send 3D reference displays information SEI

# Multiview acquisition SEI must be enabled to send this SEI

SEI\_3DReferenceDisplaysFile = 3dv\_reference\_display.cfg

**3DVCoding** specifies the compatibility category of the resulting bitstream. Value 0 specifies that resulting bitstream complies to MVC+D profile, and value 1 specifies that resulting bitstream complies to 3D-AVC profile. Default value is 1 if not present.

**NumberOfViews** specifies number of viewpoints to be coded for texture views. Note that the number of coded texture views can be different from the number of coded depth views. Typically, this parameter can be set equal to 2, or 3. Default value is 3 if not present.

**3DVCodingOrder** is a string, specifying the coding order of MVD views and components (texture and depth). Letter 'T' indicates a texture view component, letter 'D' indicates a depth view component. A number following a letter 'T' or 'D' indicates the viewpoint index in coding order starting from 0.

For example, "T0D0D1D2T1T2" specifies the following coding order: 1) texture from viewpoint 0; 2) depth from viewpoint 0; 3) depth from viewpoint 1; 4) depth from viewpoint 2; 5) texture from viewpoint 1; and 6) texture from viewpoint 2. This example was used in the CTC test condition, and is often referenced as depth first coding.

"T0D0T1D1T2D2" specifies the following coding order: 1) texture from viewpoint 0; 2) depth from viewpoint 0; 3) texture from viewpoint 1; 4) depth from viewpoint 1; 5) texture from viewpoint 2; and 6) depth from viewpoint 2. This example is often referenced as texture first coding.

**3DVConfigFile** specifies a file name for view-specific configuration which provides parameter settings for coding the texture views. See section 3.1.1 for more details.

The following parameters are relevant if 3DVCoding is set to 1:

**DepthBasedMVP** specifies whether to enable Depth based MVP (DMVP) for dependent texture view. Parameter set to 0 disables DMVP, parameter set to 1 enables DMVP. Parameter cannot be set equal to 1 if 3DVCoding is equal to 0. Default value is 0 if not present.

**SliceHeaderPred** specifies whether to enable slice header prediction. Value 0 indicates the tool is turned off. Value 1 indicates the tool is turned on. Parameter cannot be set equal to 1 if 3DVCoding is equal to 0. Default value is 0 if not present.

**PredSliceHeaderSrc, PredRefListsSrc, PredWeightTableSrc, and PredDecRefPicMarkingSrc** specifies slice header prediction related parameters. See the semantics of the syntax elements in slice header in the specification for details.

**VSP\_Enable** specifies whether to enable VSP coding tool for dependent texture view. Parameter set to 0 disables VSP, parameter set to 1 enables VSP. Parameter cannot be set equal to 1 if 3DVCoding is equal to 0. Default value is 0 if not present.

**AaptiveLuminanceCompensation** specifies whether to enable ALC coding tool for dependent texture view. Parameter set to 0 disables ALC, parameter set to 1 enables ALC. Parameter cannot be set equal to 1 if 3DVCoding is equal to 0. Default value is 0 if not present.

**AlcSearchRangeX** and **AlcSearchRangeY** specifies ALC coding tool's parameters.

**RLESkip** specifies whether to enable RLESkip coding tool for dependent texture view. Parameter set to 0 disables RLESkip, parameter set to 1 enables RLESkip. Parameter cannot be set equal to 1 if 3DVCoding is equal to 0. Default value is 0 if not present.

# Additional texture coding parameters

View-specific parameters for texture is specified in a file named by parameter 3DVConfigFile, such as view coding order, input sequence file name, etc. An example of such parameters for texture is shown below.

Table Additional Texture Configurations – PIP interview prediction

ViewCodingOrder 3-1-5

ViewId 3

InputFile balloons\_3.yuv

ViewId 1

InputFile balloons\_1.yuv

QPOffset 4

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 3

BwdAnchorRefs 0

FwdNonAnchorRefs 1 3

BwdNonAnchorRefs 0

ViewId 5

InputFile balloons\_5.yuv

QPOffset 4

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 3

BwdAnchorRefs 0

FwdNonAnchorRefs 1 3

BwdNonAnchorRefs 0

**ViewCodingOrder** specifies the coding order of different views. It is composed of a string of view numbers. The view appearing first in the string is coded first.

Following the ViewCodingOrder, there is one section for each texture view describing view-specific configurations, including inter-view dependencies. The first section is recommended for base view section.

**ViewId** specifies the view number for current view section, which should appear at the first place of the current view section.

**InputFile** specifies the input file name of the current view of this view section.

**QPOffset** specifies the QP offset of the current view relative to the base view.

**ReorderAtAnchor** specifies ordering of temporal (set by 0) and inter-view (set by 1) references in picture list at the Anchor picture. Example of parameter is “01”, which specifies that temporal reference pictures are placed in beginning of the reference picture list, with inter-view references following them.

**ReorderAtNonAnchor** specifies ordering of temporal (set by 0), inter-view (set by 1) references in picture list at the Non-Anchor picture. Example of parameter is “01”, which specifies that temporal reference pictures are placed in beginning of the reference picture list, with inter-view references following them.

**FwdAnchorRefs** specifies the reference views in LIST0. The first number indicates the number of LIST0 references and followed by the view numbers used as reference views. e.g. "0" indicates there is no LIST0 references. "1 3" indicates there is one LIST0 reference and it is from view 3.

**BwdAnchorRefs** specifies the reference views in LIST1. The first number indicates the number of LIST1 references and followed by the view numbers used as reference views. e.g. "0" indicates there is no LIST1 references. "1 3" indicates there is one LIST1 reference and it is from view 3.

Note that the interview prediction pattern is implemented by modifying the configuration in this file (and the one corresponding to depth). In the example provided above, the interview prediction is called "PIP", where the center view is coded as Intra view, the left and right views are coded as "P" views, predicted from the center view.

In the next example, the interview prediction is known as "IBP", where the left view is coded as Intra view, the right view is coded as "P" view and the central view is coded as "B" view.

Table Additional Texture Configurations – IBP interview prediction

ViewCodingOrder 1-5-3

ViewId 1

InputFile balloons\_1.yuv

ViewId 5

InputFile balloons\_5.yuv

QPOffset 4

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 1

BwdAnchorRefs 0

FwdNonAnchorRefs 1 1

BwdNonAnchorRefs 0

ViewId 3

InputFile balloons\_3.yuv

QPOffset 4

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 1

BwdAnchorRefs 1 5

FwdNonAnchorRefs 1 1

BwdNonAnchorRefs 1 5

In another example, the interview prediction "IPP", where the left view is coded as Intra view; the central view is coded as "P" view which is predicted from the left view; the right view is coded as "P" view which is predicted from the central view and left view with both being putting in LIST0.

Table Additional Texture Configurations – IPP interview prediction

ViewCodingOrder 1-3-5

ViewId 1

InputFile balloons\_1.yuv

ViewId 3

InputFile balloons\_3.yuv

QPOffset 4

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 1

BwdAnchorRefs 0

FwdNonAnchorRefs 1 1

BwdNonAnchorRefs 0

ViewId 5

InputFile balloons\_5.yuv

QPOffset 4

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 2 3 1

BwdAnchorRefs 0

FwdNonAnchorRefs 2 3 1

BwdNonAnchorRefs 0

# Depth coding parameters

Similar to texture views, depth views also have some major parameters, which are listed in the following table.

Table Main Depth Configurations

# Highest level controls

#

NumberOfViews = 3 # Number of views to encode

ForceYUV400 = 1 # 1: Code depth with 400 format. 0: Code depth with 420 format

3DVConfigFile = 3dv\_depth\_balloons.cfg # Depth config file

# Normative parameters for 3D-AVC only

#

# Depth resolution related parameters in seq\_parameter\_set\_3davc\_extension()

depth\_hor\_mult\_minus1 = 0

depth\_ver\_mult\_minus1 = 0

depth\_hor\_rsh = 1

depth\_ver\_rsh = 1

# Depth cropping related parameters in seq\_parameter\_set\_3davc\_extension()

depth\_frame\_cropping\_flag = 0

depth\_frame\_crop\_left\_offset = 0

depth\_frame\_crop\_right\_offset = 0

depth\_frame\_crop\_top\_offset = 0

depth\_frame\_crop\_bottom\_offset = 0

# Normative parameters for 3D-AVC, but non-normative for MVC+D

#

# Depth representations

NonlinearDepth = 1

NonlinearDepthModel = "2;4;7;8;10;12;14;16;17;19;20;21;22;23;24;25;26;26;27;27;27;27;27;27;26;26;25;24;23;22;20;19;17;15;13;11;9;6;3"

NonlinearDepthThreshold = 50

# Encoder controls

#

ViewSynRDO = 1

ViewSynCfg = "[ C L R ; 0.25 0.5 0.75 ]" # 3-view case, default : [ C C C ; 0.5 ]

#ViewSynCfg = "[ L R ; 0.25 0.5 0.75 ]" # 2-view case, default : [ C C ; 0.5 ]

VSDWeight = 4

SSEWeight = 1

NormalizeDepth = 1 # 0: Output depth resolution is kept the same as its input resolution

# 1: Output depth resolution is aligned with texture resolution

PostDilation = 1 # 0: No dilation for depth

# 1: Dilation for depth when writing the output YUV files

# SEIs

# Multiview Acquisition

PrecFocalLength = 18 # Specifies the exponent of the maximum allowable truncation

# error for focal\_lenth\_x and focal\_length\_y as given by 2^(-

# PrecFocalLength).(Default:18)

PrecPrincipalPoint = 18 # Specifies the exponent of the maximum allowable truncation error

# for principal\_point\_x and principal\_point\_y as given by 2^(-

# PrecPrincipalPoint).(Default:18)

PrecTranslation = 18 # Specifies the exponent of the maximum allowable truncation error

# for translation as given by 2^(-PrecTranslation).(Default:18)

MantissaLenDepthRange = 25 # Specifies the mantissa length of depth range(Default:25)

**NumberOfViews** specifies number of viewpoints to be coded for depth views. Not that the number of coded texture views can be different from the number of depth views. This parameters is typically set equal to 2, or 3. Default value is 3 if not present.

**ForceYUV400** specifies the coding format for the depth component. Value 0 means YUV420 format is used for depth coding. Value 1 indicates YUV400 format is used for depth coding. Default is 1 if not present.

**3DVConfigFile** specifies a file name for view-specific configuration which provides parameter settings for coding of depth views. See section 3.2.1 for more details.

**NonlinearDepth** specifies the way to represent depth values. Value 1 indicates to enable nonlinear depth representation. Value 0 indicates to disable nonlinear depth representation. Default value is 0 if not present. NonlinearDepth tool can be activated in both MVC+D and 3D-AVC complied bitstreams. In the case of 3D-AVC this is a normative tool, in the case of MVC+D this is non-normative tool and it is signaled through SEI.

**NonlinearDepthModel** specifies the nonlinear depth model when NonlinearDepthis equal to 1. For example, "10;19;24;27;26;22;13" as deviations from diagonal line 0..255.

**NonlinearDepthThreshold** specifies the nonlinear depth threshold when NonlinearDepthis equal to 1. The parameter equal to non-zero would automatically turn on/off the Nonlinear Depth Representation (NDR) tool based on average depth value in first depth frame of the base view. NDR is turned off if the average depth value is less than NonlinearDepthThreshold ; NDR is turned on if the average depth value is not less than NonlinearDepthThreshold . NonlinearDepthThreshold equal to 0 specifies that NDR is always turned on.

The following parameters are relevant if 3DVCoding is set to 1.**depth\_hor\_mult\_minus1, depth\_ver\_mult\_minus1, depth\_hor\_rsh,** and **depth\_ver\_rsh** specify parameters to depth-based disparity value derivation process (specified in subclause J.8.2.1.1 of the specification). When not present, depth\_hor\_mult\_minus1 and depth\_ver\_mult\_minus1 are inferred to be equal to 1, and depth\_hor\_rsh and depth\_ver\_rsh are inferred to be equal to 0. depth\_hor\_mult\_minus1 and depth\_ver\_mult\_minus1 shall be in the range of 0 to 1023, inclusive. depth\_hor\_rsh and depth\_ver\_rsh shall be in the range of 0 to 31, inclusive.

**depth\_frame\_cropping\_flag** equal to 1 specifies that the frame cropping offset parameters for depth view components follow next in the sequence parameter set. Value 0 indicates such parameters not present. Default value is 0 if not present.

**depth\_frame\_crop\_left\_offset, depth\_frame\_crop\_right\_offset, depth\_frame\_crop\_top\_offset,** and **depth\_frame\_crop\_bottom\_offset** specify the samples of the decoded depth view components in the coded video sequence that are output from the decoding process, in terms of a rectangular region specified in frame coordinates for output.

The following parameters are used to control encoder, applicable for both MVC+D and 3D-AVC bitstreams.

**ViewSynRDO** equal to 1 specifies VSO is turned on for rate distortion optimization. Value 0 indicates VSO is turned off. Default value is 0 if not present.

**ViewSynCfg** specifies the parameters on how the VSO is configured. It must be present if ViewSynRDO is equal to 1. In particular, the parameter, a string composed of letter and numbers, specifies the rendering direction and positions for view synthesis. Letters, "C","L", and "R", indicate the rendering directions of texture/depth views. Letter "L" indicates the current view would be used as reference for view synthesis for views on its left, letter "R" indicates the current view would be used as reference for view synthesis for views on its right, and letter "C" indicates the current view would be used as reference for view synthesis for views on its left and right. The first letter corresponds to the view with view index 0, the second letter corresponds to the view with view index 1, etc. In addition, numbers, e.g. 0.25, 0.5, 0.75, specify the viewpoints of the virtually rendered views.

For example, [ C L R ; 0.25 0.5, 0.75 ] specifies that texture/depth views from viewpoint 0 are used as reference to render a virtual view on its left and right (C), texture/depth views from viewpoint 1 are used as reference to render a virtual view on its left (L), and texture/depth views from viewpoint 2 are used as reference to render a virtual view on its right. The target virtual viewpoints are positioned at 0.25, 0.5 and 0.75 between two neighboring texture/depth views that are coded.

**VSDWeight** further specifies the weighting factor of view synthesis distortion in the RDO optimization. Default value is 4 if not present.

**SSEWeight** further specifies the weighting factor of SSE distortion in the RDO optimization. Default value is 1 if not present.

The following parameters are used to control post-processing of depth map, applicable for both MVC+D and 3D-AVC bitstreams. Note, post-processing is disabled in conformance bitstreams production:

**NormalizeDepth** equal to 0 specifies the depth is outputted at the same resolution as its input resolution. Value 1 specifies that the depth is output at the same resolution as the accompany texture view. Default value is 0 if not present.

**PostDilation** equal to 1 specifies that a dilation post processing is applied to depth before it is outputted. Value 0 specifies that no such dilation is applied. Default value is 0 if not present.

# Additional depth coding parameters

Further view-specific parameters for depth is specified in a file named by parameter 3DVConfigFile, such as view coding order, input sequence file name, etc. An example of such parameters for texture is shown below.

Table Additional Depth Configurations – PIP interview prediction

# Overview

ViewCodingOrder 3-1-5

CameraOrder 1-3-5

CameraParameterFile cam\_param\_balloons.txt

AcquisitionIdc 1

# View section 3

ViewId 3

InputFile depth\_balloons\_0.5\_3.yuv

ZNear 448.251214

ZFar 11206.28035

CameraName param\_balloons3

TanslationFile NULL

ZNearFile NULL

ZFarFile NULL

GridPosX 0

GridPosY 0

# View section 1

ViewId 1

InputFile depth\_balloons\_0.5\_1.yuv

QPOffset 0

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 3

BwdAnchorRefs 0

FwdNonAnchorRefs 1 3

BwdNonAnchorRefs 0

ZNear 448.251214

ZFar 11206.28035

CameraName param\_balloons1

TanslationFile NULL

ZNearFile NULL

ZFarFile NULL

GridPosX 0

GridPosY 0

# View section 5

ViewId 5

InputFile /data/3dtv/3dv\_cfp\_seqs/LowRes/depth\_balloons\_0.5\_5.yuv

QPOffset 0

ReorderAtAnchor 01

ReorderAtNonAnchor 01

FwdAnchorRefs 1 3

BwdAnchorRefs 0

FwdNonAnchorRefs 1 3

BwdNonAnchorRefs 0

ZNear 448.251214

ZFar 11206.28035

CameraName param\_balloons5

TanslationFile NULL

ZNearFile NULL

ZFarFile NULL

GridPosX 0

GridPosY 0

**ViewCodingOrder** specifies the coding order of different views. It is composed of a string of view numbers. The view appearing first in the string is coded first.

**CameraOrder** specifies view orders appearing from the left to the right. It is composed of a string of view numbers. The view appearing first in the string is on the leftmost.

**CameraParameterFile** specifies the camera parameter file. The example of camera parameter file is given in Appendix 1.

**AcquisitionIdc** specifies a type of depth representation approach. Parameters being set to 0 specifies that depth represents the disparity information. Parameter being set to 1 specifies that depth represents the distance information measured in a 3D coordination system, at the same time, all pictures within one view share the same camera and depth range information. Parameter being set to 2 specifies that the camera and depth range information may vary from picture by picture. Picture-varying camera and dynamical range parameters to be specified in external files through parameters TanslationFile, ZNearFile and ZFarFile.

Next, there is one view section for each view to be coded. The first section is recommended for the base view.

**ViewId** specifies the view number for current view section, which should appear at the first place of the current view section.

**InputFile** specifies the input file name of the current view. The first view section corresponds to the base view.

**QPOffset** specifies the QP offset of the current view relative to the base view.

**ReorderAtAnchor** specifies ordering of temporal (set by 0) and inter-view (set by 1) references in picture list at the Anchor picture. Example of parameter is “01”, which specifies that temporal reference pictures are placed in beginning of the reference picture list, with inter-view references following them.

**ReorderAtNonAnchor** specifies ordering of temporal (set by 0), inter-view (set by 1) references in picture list at the Non-Anchor picture. Example of parameter is “01”, which specifies that temporal reference pictures are placed in beginning of the reference picture list, with inter-view references following them.

**FwdAnchorRefs** specifies the reference views in LIST0. The first number indicates the number of LIST0 references and followed by the view numbers used as reference views. e.g. "0" indicates there is no LIST0 references. "1 3" indicates there is one LIST0 reference and it is from view 3.

**BwdAnchorRefs** specifies the reference views in LIST1. The first number indicates the number of LIST1 references and followed by the view numbers used as reference views. e.g. "0" indicates there is no LIST1 references. "1 3" indicates there is one LIST1 reference and it is from view 3.

**ZNear** and **ZFar** specify the nearest and farthest Z values. Together with camera parameters, they are used for conversion between depth value and disparity value.

**CameraName** specifies the camera name used to indicate the current view in the camera parameter file.

**TanslationFile**, **ZNearFile** and **ZFarFile** specify the file names of translation parameter, ZNear and Zfar parameter if **AcquisitionIdc** is set equal to 2. Each of these parameters are are provided on a basis of frames. The example of such parameter file is given in Appendix 2, Appendix 3 and Appendix 4.

**GridPosX** specifies, for the current view, a horizontal offset of a depth sampling grid relative to the luma texture sampling grid in texture luma sample units.

**GridPosY** specifies, for the current view, a vertical offset of a depth sampling grid relative to the luma texture sampling grid in texture luma sample units.

Note that the example provided above corresponds to "PIP" interview prediction. Similar modifications as in texture (see Table 4 and Table 5) could be made to support "IBP" and "IPP" interview predictions.

# Configuration parameters for interlace coding

Note: This subsection is composed mainly based on information provided by Ching-Chieh Lin from ITRI.

The main changes of the configurations for interlace coding for interlace texture and depth are summarized as in Table 8 and Table 10, respectively.

Table Typical texture configuration parameters for interlace texture

ProfileIDC = 128

LevelIDC = 41

NumberReferenceFrames = 2

NumberOfViews = 2

3DVCodingOrder = "T0D0T1D1"

NumberBFrames = 3

ExplicitHierarchyFormat = "B1r5t1B0e6t2B2e6t2" #GOPSize 4

ReferenceReorder = 0

PicInterlace = 1

Due to the huge memory used with 1088P sequences, the GOP size may be quite limited. In the above example, GOP size is set equal to 4 and NumberBFrame is set equal to 3 to limit the required memory.

For 1024x768 sequences, GOP can be set to 8 and NumberBFrames set to 7, as in Table 9 .

Table Typical texture configuration parameters for interlace texture – 1024x768 sequences

NumberBFrames = 7

ExplicitHierarchyFormat = "B3r4t1B1r5t2B0e6t3B2e6t3B5r5t2B4e6t3B6e6t3"

Table Typical depth configuration parameters for interlace depth

LevelIDC = 41

NumberReferenceFrames = 2

NumberOfViews = 2

ViewSynRDO = 0

ReferenceReorder = 0

PicInterlace = 1

Example configuration files are provided under ./configs/mvc+d\_interlace in the software package.

# Decoder run-time configuration parameters

All JM decoder configuration parameters are reused. Especially, parameter DecodeAllLayers equal to 0 specifies that only the texture base view is to be decoded. DecodeAllLayers equal to 1 specifies that all coded texture and depth view components are to be decoded.

In addition to JM decoder configuration parameters, there are a few new parameters as listed in the following table.

Table Decoder Configurations

#

DecodeAllLayers = 1 # decode all view and all components

OutputRecYUVFile = 1 # output YUV reconstructed YUV files

OutputFileForTexture = "texture\_dec.yuv" # Output file, YUV/RGB

WriteUVForTexture = 1 # 1: write UV components for texture; 0: not writing

OutputFileForDepth = "depth\_dec.yuv" # Output file, YUV/RGB

WriteUVForDepth = 1 # 1: write UV components for depth; 0: not writing

NormalizeDepth = 1 # 0: Output depth resolution is kept the same as its input resolution

# 1: Output depth resolution is aligned with texture resolution

PostDilation = 1 # 0: No dilation for depth

# 1: Dilation for depth when writing the output YUV files

CameraFile = "camera.txt" # Output file, camera parameters

3DReferenceDisplaysOutputFile = "ReferenceDisplaySEIOutput.txt"

**OutputRecYUVFile** equal to 1 specifies that the decoded YUV are to be outputted to files. Value 0 specifies that no YUV files from the decoding will be outputted. Default value is 0 if not present.

**OutputFileForTexture** specifies the file name base of texture YUV files. The view index will be appended to the end of the file name.

**WriteUVForTexture** equal to 1 specifies that the chroma components will be written to the YUV output file. Value 0 specifies that no chroma components will be written to the YUV output file when the texture is coded in YUV400 format. Note that if the texture is coded with non-YUV400, this parameter does not have impacts and the chroma components are always outputted. Default value is 1 if not present.

**OutputFileForDepth** specifies the base file name of depth YUV files. The view index will be appended to the end of the file name.

**WriteUVForDepth** equal to 1 specifies that the chroma components will be written to the YUV output file. Value 0 specifies that no chroma components will be written to the YUV output file when the depth is coded in YUV400 format. Default value is 0 if not present.

**CameraFile** specifies the file name used to store the outputted camera parameters. It may be provided as input for a view synthesis software to render a virtual view.

The following parameters are used to control post-processing of depth map, applicable for both MVC+D and 3D-AVC bitstreams. Note, post-processing is disabled in conformance bitstreams production:**NormalizeDepth** equal to 0 specifies the depth is outputted at the same resolution as its input resolution. Value 1 specifies that the depth is output at the same resolution as the accompany texture view. Default value is 0 if not present.

**PostDilation** equal to 1 specifies that a dilation post processing is applied to depth before it is outputted. Value 0 specifies that no such dilation is applied. Default value is 0 if not present.

# Appendix 1: Example of camera parameter file

Following is an example of camera parameter file.

Table Example camera file

param\_balloons00  
2241.25607 0.0 701.5  
0.0 2241.25607 514.5  
0.0 0.0 1.0   
0.0   
0.0   
1.0 0.0 0.0 0.0  
0.0 1.0 0.0 0.0   
0.0 0.0 1.0 0.0

param\_balloons01  
2241.25607 0.0 701.5  
0.0 2241.25607 514.5  
0.0 0.0 1.0   
0.0   
0.0   
1.0 0.0 0.0 5.0  
0.0 1.0 0.0 0.0   
0.0 0.0 1.0 0.0

param\_balloons02  
2241.25607 0.0 701.5  
0.0 2241.25607 514.5  
0.0 0.0 1.0   
0.0   
0.0   
1.0 0.0 0.0 10.0  
0.0 1.0 0.0 0.0   
0.0 0.0 1.0 0.0

# Appendix 2: Example of translational parameter file

Following is an example of translational parameter file, composing of translation parameters for frames 0, 1 and 2.

Table Example translational file

2824.7961840820300  
2796.7812475585900  
2768.8285180664000

# Appendix 3: Example of ZNear parameter file

Following is an example of ZNear parameter file, composing of ZNear parameters for frames 0, 1 and 2.

Table Example ZNear file

70625  
69812.5  
68937.5

# Appendix 4: Example of ZFar parameter file

Following is an example of Zfar parameter file, composing of ZFar parameters for frames 0, 1 and 2.

Table Example ZFar file

662000  
663000  
664500