INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11 MPEG2012/ M25013
May 2012, Geneva, Switzerland

Source **Poznań University of Technology,**
Chair of Multimedia Telecommunications and Microelectronics, Poznań, Poland
Status **Report**
Title **Report on progress of integration of Poznan tools in 3D-HTM software**
Author **Krzysztof Wegner** (kwegner@multimedia.edu.pl)

Table of contents

1 Introduction

This documents reports progress on integration of coding tools that originate from compression technology proposed by Poznan University of Technology [1] in response to Call for Proposals on 3D Video Coding Technology [2].

2 Overview

All of the integrated tools are marked in the source code with define labels with "POZNAN_" prefix, and can be switch off at compilation time. Integration process for those tools is discussed in further sections:

- View synthesis based inter-view disocclusion prediction (Section 3.1),
- Depth-based motion parameter prediction (Section 3.2),
- Adjustment of QP of texture based on depth data (Section 3.3),
- Non-linear depth representation (Section 3.4),

Moreover all of the tools was adjusted in order to work with Flexible Coding Order implemented by Nokia [x]

3 Poznan tools description

3.1 View synthesis in encoder and decoder

Many tools utilize view synthesis algorithm or synthesized view based on some previously coded view. We have implemented frame work for view synthesis algorithm to be operation at the encoder and decoder side and available to use for various tools. This reduce complexity and allow to use single consistent view synthesis algorithm for all tools in 3D-HTM.

Moreover duo to Flexible Coding Order camera parameter which was are sended in texture SPS now can be transmitted in depth SPS as well.

3.1.1 Tool activation

The tool can be enabled or disabled in "TypeDef.h" file with `POZNAN_SYNTH and POZNAN_AVAIL` flag.

3.1.2 Implementation details

We have added image buffer in `TComPic` class for image synthesized form base view along with `getPicYuvSynth()` method to obtain it. Before encoding or decoding each view, encoder and decoder synthesized coded view base on base view and store in this buffer.

3.2 View synthesis based inter-view disocclusion prediction

With this tool, view synthesis is used as a primary inter-view prediction mechanism. The encoder and the decoder use the single view extrapolation algorithm, existing in 3D-HTM software. Basing on all already coded views, a new virtual view is synthesized in the position of the currently coded view. During extrapolation of virtual view some regions of image are being disoccluded. Those disoccluded regions are identified and marked on a binary map, called availability map, which controls coding and decoding process. The coder and decoder simultaneously use this map to determine, whether given CU is coded or not. Because in a typical case most of the scene is the same in all of views, only small parts are disoccluded in subsequently coded views, and thus only small amount of CUs is directly encoded, rest of the CUs are synthesized at the decoder.

Current encoder implementation use this tools for both texture and depth component. Some problems with VSO have been observed and need further harmonization.

3.2.1 Tool activation

The tool can be enabled or disabled in "TypeDef.h" file with `POZNAN_ENCODE_ONLY_DISOCCLUDED_CU` flag.

3.2.2 Tool dependences

The view synthesis based inter-view prediction tool requires synthesis algorithm operation at the decoder. This can be enabled with `POZNAN_SYNTH` and `POZNAN_AVAIL` flag.

3.2.3 Implementation details

Additional buffers to keep availability maps were implemented in `TEncCU` class and in `TComPic` class. Availability maps are computed in `TRenTop::extrapolateAvailabilityView (...)` function which is slightly modified `TRenTop::extrapolateView(...)` function from 3D-HTM base software.

In encoding/decoding process encoder/decoder determine if block should be encoded into bitstream/decode from bitstream using `TComPic::checkSynthesisAvailability(...)` function.

3.2.4 Configuration parameters

This tools can be enabled and disabled from configuration file by “CU-Skip” option.

3.3 Depth-based motion parameter prediction

Proposed Depth-based motion parameter prediction (DBMP) is a coding tool for multiview video coding which originates from the idea that motion fields of neighboring views in multiview sequence are highly correlated. In the method, the motion information, such as motion vectors and reference indices, for each pixel of encoded coding unit (CU) is directly inferred from already encoded CUs in the neighboring views at the same temporal instance (Fig. 1). This procedure is repeated independently for every pixel of encoded CU. Consequently, motion vectors and reference indices for CU are not transmitted in a bitstream but obtained from the reference view.

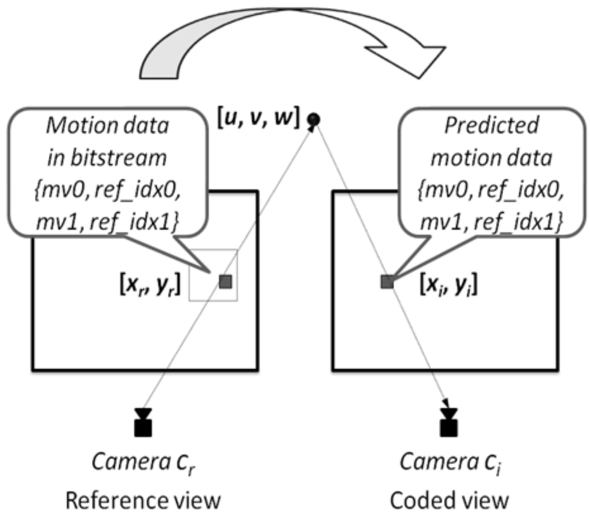

Fig. 1. Independent derivation of motion information for each point of encoded CU from corresponding point in reference view [5].

The DBMP motion prediction has been implemented as an extension to merge candidate list used in HEVC. Apart from existing merge candidates used to predict motion information by the HEVC codec, the DBMP candidate was added as new predictor on the merge candidate list. This new DBMP predictor is available (inserted to the list) only for CUs in non-anchor pictures of non-base views. As a result, base view remains HEVC-compatible.

3.3.1 Tool activation

The tool can be enabled or disabled in "TypeDef.h" file with `POZNAN_DBMP` flag.

Moreover this tool can be disabled independently for texture and depth coding using `POZNAN_DBMP_USE_FOR_TEXTURE` and `POZNAN_DBMP_USE_FOR_DEPTH` flags in "TypeDef.h" respectively.

3.3.2 Tool dependences

The tool requires activation of `POZNAN_MP` flag in "TypeDef.h" file.

Position of the DBMP candidate in the merge candidate list is determined by `POZNAN_DBMP_MERGE_POS` in "CommonDef.h" file. This setting overwrites `PDM_MERGE_POS` setting and, also, is overwritten by `HHI_MPI_MERGE_POS` setting.

3.3.3 Implementation details

The functionality of DBMP prediction is implemented in `TComMP` class used to predict information from already encoded frames from neighboring views into currently coded frame. The inter-view prediction algorithm is implemented in `TComMP:: pairMultiview (...)` function. The function uses extra data buffers to save reference position of the inter-view predicted reference CUs for each point of encoded frame.

The process of encoding CU using the DBMP candidate is implemented in `TComDataCU::getInterMergeCandidates(...)` and `TComPrediction::motionCompensation_DBMP(...)` functions.

In general, this tool has low computational complexity, because mainly consists of once-per-frame inter-view prediction processing and extra merge candidate checking for each encoded CU.

3.3.4 Configuration parameters

No parameters for this tool needs to be specified in configuration file. However, the tool can be deactivated from the configuration file using "`DBMP`" label set to “0” (zero).

3.4 Adjustment of QP of texture based on depth data

In order to improve perceptual quality of coded texture, a tool for bit assignment in the texture layer has been developed. The basic idea is to increase texture quality of objects in the foreground and to increase compression factor (decrease texture quality) for objects in the background. The quality is adjusted at coding units (CUs) level with use of quantization parameter QP that depends on the corresponding depth values. The QP adjustment is done simultaneously in coder and decoder so that no additional information are send. Described tool is disabled in the base view to preserve HEVC compatibility.

3.4.1 Tool activation

The tool can be enabled or disabled in "TypeDef.h" file with `POZNAN_TEXTURE_TU_DELTA_QP_ACCORDING_TO_DEPTH` flag.

Moreover as this tool can be enabled in the base view to enhance performance by `POZNAN_TEXTURE_TU_DELTA_QP_NOT_IN_BASE_VIEW` flag in “TypeDef.h”

3.4.2 Tool dependences

If Flexible Coding Order is disabled, 3D-HTM software uses Depth-after-Texture coding order. In that case implemented tool is using depth map synthesized from previously coded and reconstructed depth maps.

In other case, if Flexible Coding Order tool is enabled, there is a chance that a depth map was coded before a texture for given view. In that situation this reconstructed depth map is used instead of synthesized depth map.

In order to enable this tool synthesis tool needs to be enabled too (`POZNAN_SYNTH` flag in "TypeDef.h" file).

3.4.3 Implementation details

No actual QP value update in coding unit object fields is done. QP value is adjusted through a parameter of class function `TComTrQuant::setQPforQuant(...)`. Functions used for computing QP adjustment are implemented in `TComDataCU` class.

3.4.4 Configuration parameters

This tool can be enabled from configuration file by option “TDdQP”

3.5 Non-linear depth representation

M22697 describes a normative tool named non-linear depth representation. The depth is internally represented in such a way that the closer objects are represented more accurately than distant ones.

In the original proposal, internal depth sample values were defined by the following power-law expressions, similar as in the case of well known gamma correction:

$$depth\ value\ internal = \left(\frac{depth\ value\ external}{maximum\ value\ external}\right)^{exponent} \cdot maximum\ value\ internal$$

Where "maximum value" field represents range of used disparity values.

Such approach unfortunately cannot be seamlessly implemented with integer-only operations.
The exact shape of curves for non-linear representation is defined by means of line-segment-approximation. The first (0,0) and the last (255,255) nodes are predefined. Additional nodes can be transmitted in SPS in form of deviations from straight-line curve (linear representation).
Nodes are defined in configuration file with "NonlinearDepthModel" parameter.
Additionally, this allows experimentation with other non-linear curves that the above-mentioned exp-based curve.

In most of places, linear-to-non-linear conversion is done through LUT tables, and in some cases (view synthesis etc) non-linearity has been incorporated into existing LUT tables.

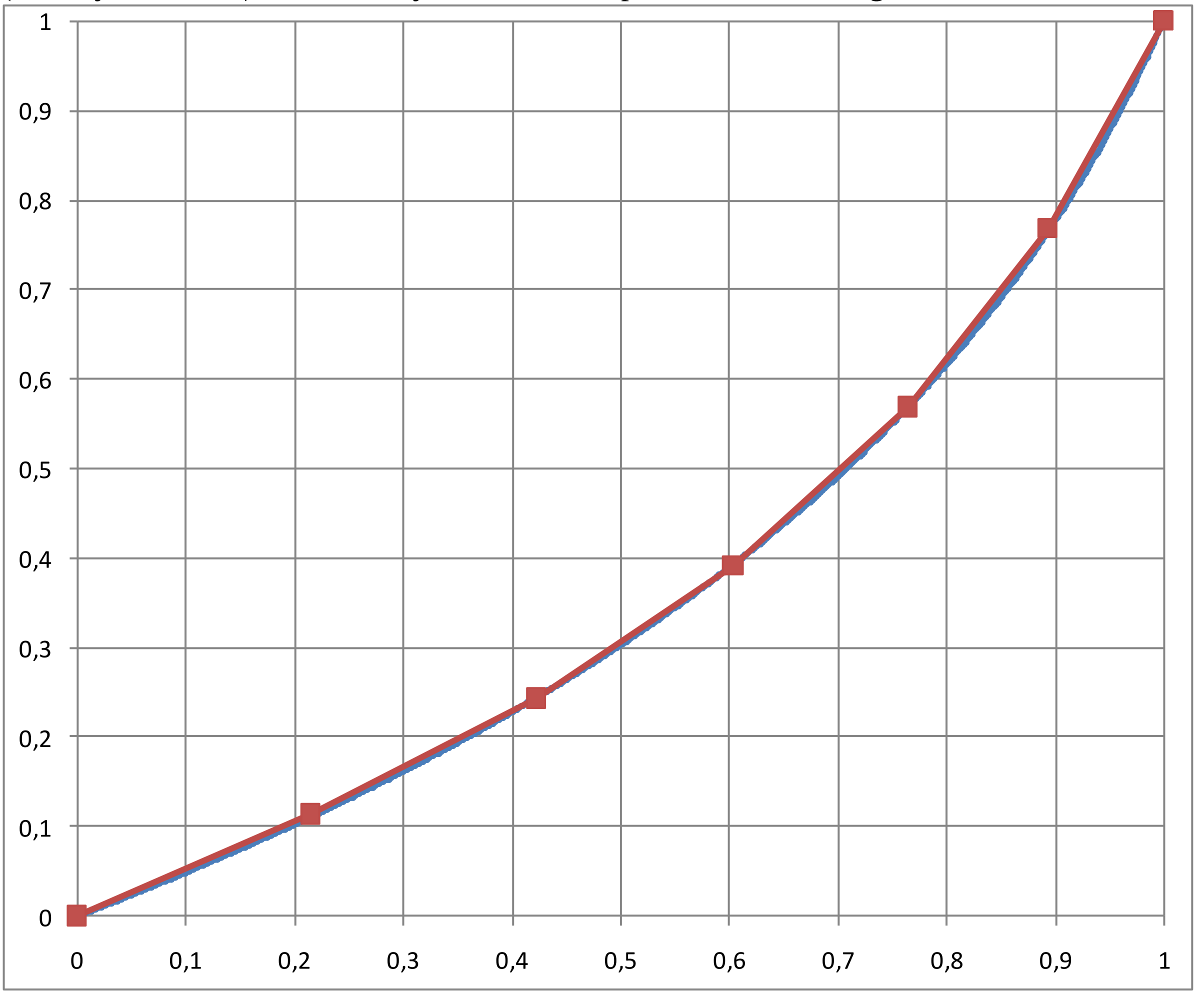

Fig. 2. An exemplary approximation of non-linear curve with 5 values being sent in SPS.

As some experiments has shown, Non-linear Depth Representation gives subjective gains but sometimes imposes objective losses. The objective losses can be reduced if this tool is turned off in some sequences. Specifically, this tools is turned off, if weighted average of depth map of the center view (the center of the scene) is in low disparity ranges. The selection is done automatically with use of the following value:

$$center\_disparity = \frac{\sum_{i=0..255} disparity\_histogram(i) \cdot i}{\sum_{i=0..255} disparity\_histogram(i)}$$

where $disparity\_histogram(i)$ is a histogram of the first frame of depth map in center view.

If $center\_disparity$ (ranging from 0 to 255) is lesser that some threshold (e.g. 100), then Non-linear Depth Representation tool is turned off.
The rationale behind such approach is that low disparity values are highly distorted.
Threshold for $center\_disparity$ can be set with "NonlinearDepthThreshold" parameter.
Of course NonlinearDepthThreshold=0 means "always on" – NDR is never disabled.

3.5.1 Tool activation

The tool can be enabled or disabled in "TypeDef.h" file with `POZNAN_NONLINEAR_DEPTH` flag.

3.5.2 Tool dependences

This tool requires no additional tools activation.

3.5.3 Implementation details

Function `TComPicYuv::power(...)` implemented in `TComPicYuv` class is used to transform `TComPicYuv` object to internal or external representation. This function uses `TComPowerConverter` class object that convert individual internal and external values.

With the object of correct work of view synthesis algorithm, already implemented in the base software, minor changes were made. Look-Up Tables (LUTs) used in this synthesis algorithm were changed to obtain proper results with non-liner depth maps. It is done with modifications in `TAppComCamPara::xSetShiftParametersAndLUT(...)` function. LUTs precision was also increased (enabled with `POZNAN_LUT_INCREASED_PRECISION` flag in "CommonDef.h" file), because originally did not supported Internal Bit Depth Increase (IBDI) tool of the codec.

Everywhere in codec, depth maps are kept in transformed (non-linear) form. To obtain an original depth map values reverse transformation is needed. While writing a depth map to YUV file this reverse transformation is done automatically (function `TAppDecTop::xWriteOutput(...)` and `TAppEncTop::xWriteOutput(...)` were modified. View synthesis and View Synthesis Optimization (VSO) tools are patched with use of modified LUTs. In general, this tool has no computational complexity, because mainly consists of pre-/post-processing of the coded depth maps.

3.5.4 Configuration parameters

In configuration file, the tool can be turned on with the following exemplary configurations:

a) 7-node non-linear curve approximation

```
NonlinearDepth: 1
NonlinearDepthModel: 10 19 24 27 26 22 13
NonlinearDepthThreshold: 100
```

b) 40-node non-linear curve approximation

```
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: 100
```

4 Conclusions

All tools have been already integrated in the 3D-HTM software. Integrated version marked as **3D-HTM tag 0.5** can be found at:

https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/0.5/

All coding tools was tested by HHI and results are in perfect match. HHI also confirm that there are **no problems** with the source code, implementation or simulations of all integrated tools.

5 Acknowledgement

This work was supported by the public funds as a research project.

6 Patent rights

Poznan University of Technology may have IPR relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard (per box 2 of the UTI-T/ITU-R/ISO/IEC patent statement and licensing declaration form).

7 References

[1] M. Domański, T. Grajek, D. Karwowski, K. Klimaszewski, J. Konieczny, M. Kurc, A. Łuczak, R. Ratajczak, J. Siast, O. Stankiewicz, J. Stankowski, K. Wegner, "Technical Desciption of Poznan University of Technology proposal for Call on 3D Video Coding Technology", ISO/IEC JTC1/SC29/WG11 (MPEG) M22697, Geneva, Switzerland, November 2011.

[2] "Call for Proposals on 3D Video Coding Technology", ISO/IEC JTC1/SC29/WG11 MPEG2011/N12036, Geneva, Switzerland, March 2011

[3] H. Schwarz, K. Wegner, "Test Model under Consideration for HEVC based 3D video coding", ISO/IEC JTC1/SC29/WG11 (MPEG) w12350, Geneva, Switzerland, November 2011.

[4] https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/

[5] J. Konieczny, M. Domański, "Inter-View Direct Mode for Multiview Video Coding", ISO/IEC JTC1/SC29/WG11, MPEG2010/M17800, Geneva, Switzerland, July 2010.

[6] J. Konieczny, M. Domański, "Extended Inter-View Direct Mode for Multiview Video Coding", ICASSP 2011, Prague, Czech Republic, May 2011.