INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC 1/SC 29/WG 7 MPEG 3D GRAPHICS CODING

ISO/IEC JTC 1/SC 29/WG 7 nXXXXX
September 2022, Online

Title:	Video-based dynamic mesh coding test model v2.0 user manual	
Source:	WG 7, MPEG 3D Graphics Coding	
Purpose:	User manual	
Status:	Draft	
Author(s):	Julien Ricard	*julien.ricard@interdital.com*
	Wenjie Zou	*wjzou@xidian.edu.cn*

Video-based dynamic mesh coding test model v2.0 user manual

Abstract

This document is a user manual describing usage of reference software for the V-DMC project. It applies to version 2.0 of the software.

Contents

1 General Information

Reference software is being made available to provide a reference implementation of the video-based dynamic mesh coding standard being developed by MPEG-3DG (ISO/IEC SC29 WG7).

One of the main goals of the reference software is to provide a basis upon which to conduct experiments in order to determine which coding tools provide desired coding performance. It is not meant to be a particularly efficient implementation of anything, and one may notice its apparent unsuitability for a particular use. It should not be construed to be a reflection of how complex a production-quality implementation of a future standard would be.

This document aims to provide guidance on the usage of the reference software. It is widely suspected to be incomplete and suggestions for improvements are welcome. Such suggestions and general inquiries may be sent to the general MPEG 3DG email reflector at `mpeg-3dgc@gti.ssr.upm.es` (registration required).

2 Obtaining the software

The authoritative location of the software is the following git repository: `http://mpegx.int-evry.fr/software/MPEG/dmc/TM/mpeg-vmesh-tm`

Each released version may be identified by a version control system tag in the form: `v2.0`

An example:

```
$ git clone \
   http://mpegx.int-evry.fr/software/MPEG/dmc/TM/mpeg-vmesh-tm.git
$ cd mpeg-vmesh-tm
$ git checkout v2.0
```

> It is strongly advised to obtain the software using the version control system rather than to download a zip (or other archive) of a particular release. The build system uses the version control system to accurately identify the version being built.

3 Building

3.1 Building script

A bash script is provided to facilitate the building operations.

To build V-DMC test model softwares with this script please use the following command line:

```
$ ./build.sh
$ ./build.sh --help
./build.sh mpeg-vmesh-tm building script:

    Usage:
       -h|--help     : Display this information.
       -o|--ouptut   : Output build directory.
       -n|--ninja    : Use Ninja.
       --debug       : Build in debug mode.
       --release     : Build in release mode.
       --doc         : Build documentation (latex and pdflatex requiered).
       --format      : Format source code.
       --tidy        : Check source code with clang-tidy.
       --cppcheck    : Check source code with cppcheck.
       --test        : Build unit tests.
       --meshType=*  : Define template mesh type: float or double.
```

```
        --codeCodecId: Code codec id used in the bitstream.

    Examples:
      ../build.sh
      ../build.sh --debug
      ../build.sh --doc
      ../build.sh --format
```

Another script could be used to clean the current solutions with the following command lines:

```
$ ./clear.sh      # Remove ./build/ sub-folder.
$ ./clear.sh all  # Remove all cloned dependencies.
```

3.2 Build manually

Standard CMake build commands can be used to build the software depending on the system you used.

3.2.1 OSX

```
$ mkdir build
$ cmake -S. -Bbuild -G Xcode
$ xcodebuild -project build/vmesh.xcodeproj -configuration Debug
```

3.2.2 Linux

```
$ mkdir build
$ cmake -DCMAKE_BUILD_TYPE=Release -S. -Bbuild/Release
$ cmake --build ./build/Release --config Release --parallel 12
```

3.2.3 Windows

```
$ md build
$ cmake -DCMAKE_BUILD_TYPE=Release -S. -Bbuild/Release
$ cmake --build ./build/Release --config Release --parallel 12
```

3.3 Dependencies

The V-DMC test model software uses several dependencies that are cloned and patched by the CMake building process.

These dependencies are:

URL	Commit/tag
Directx-headers	1b79ddaeabc4b16c772ca63adc5bdf7d5f741460
Directx-math	b404898c9dcaff7b686bbaf6d2fba8ff0184a17e
Directx-mesh	2c0ed18e271afa99a70948f784dfe082127fa0de
UVAatlas	5af1b5d2a0fd9e0e5d17aa0971ab17c890e318e0
Draco	266f47ce58b0568ff9328e12174b25cb0fbd3b2e
HDRTools	v0.23
HM	HM-16.21+SCM-8.8
VTM	VTM-13.0
mpeg-pcc-mmetric	1_0_1_lib_beta_v5

4 Architecture

V-Mesh test model software is organized as shows in figure 1.

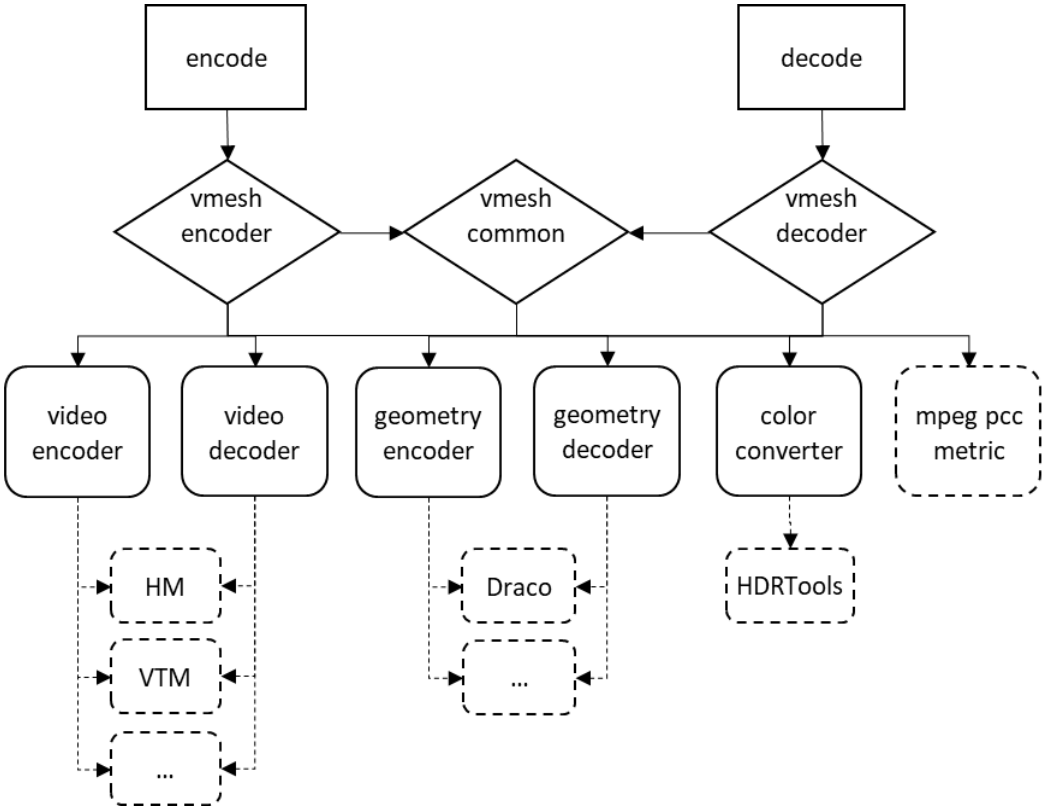

Figure 1 – Scheme of acrhitecture of V-Mesh test model.

4.1 Core libraries

The core codec processes are grouped in three lirbaries:

- `vmeshCommon` containing the util objects and the processes shared by V-Mesh encoding and decoding processes.
- `vmeshEncoder` containing the V-Mesh encoding processes.
- `vmeshDecoder` containing the V-Mesh decoding processes.

4.2 Wrapper libraries

To unify interfaces with external libraries used to encode/decode meshes, encode/decode/convert videos and compute metrics, wrapper libraries have been created:

- `videoEncoder`: wrapper to HM encoder, VTM encoder;
- `videoDecoder`: wrapper to HM decoder, VTM decoder;
- `geometryEncoder`: wrapper to Draco encoder;
- `geometryDecoder`: wrapper to Draco decoder;
- `colourConverter`: wrapper to HDRTools.

These libraries are based on a virtual object that can be derived to implement one specific interface with the external libraries. The source codes of the wrapper libraries are stored in the `source/wrapper/` sub-folder.

4.3 Applications

The source codes of V-Mesh applications are stored in the `source/app/` sub-folder.

4.3.1 main applications

The two main application of the V-Mesh test model are:

- `encode`: that can be used to encode mesh sequence fo a V-MEsh bitstream.
- `decode`: that decode V-Mesh bitstream.

The following section shows examples of the usage of these softwares.

4.3.2 Wrapper applications

To evaluate the wrapper libraries specific applications have been created. These applications can be used to crosscheck the usage of the external applications. The source code of these applications are in `source/app/wrapper/` sub-folder:

- `encodeDraco`;
- `decodeDraco`;
- `encodeVideo`;
- `decodeVideo`;
- `colourConverte`;
- `metrics`.

4.3.3 Unit test applications

To evaluate the source code and to garantee a early regression detection, unit test application has been created.

The unit test application is based on Google Testing Framework. The source code of this software is stored in `source/app/unitTests/`. The list of the unit tests that are implemented can be logged with the following command line:

```
$ ./build/Release/bin/unitTests --gtest_list_tests
draco.
  encode
  decode
metrics.
  compare
hm.
  disp
  disp2
  texture
colourConvert.
  hdrToolsUp
  hdrToolsDown
vmesh.
  all
```

The unit tests can be executed with:

```
$ ./build/Release/bin/unitTests -v 0
...
[       OK ] hm.texture (4770 ms)
[----------] 3 tests from hm (5595 ms total)

[----------] 2 tests from colourConvert
[ RUN      ] colourConvert.hdrToolsUp
[       OK ] colourConvert.hdrToolsUp (190 ms)
[ RUN      ] colourConvert.hdrToolsDown
[       OK ] colourConvert.hdrToolsDown (197 ms)
[----------] 2 tests from colourConvert (388 ms total)
```

```
[----------] 1 test from vmesh
[ RUN      ] vmesh.all
[       OK ] vmesh.all (49463 ms)
[----------] 1 test from vmesh (49463 ms total)

[----------] Global test environment tear-down
[==========] 9 tests from 5 test suites ran. (64516 ms total)
[  PASSED  ] 9 tests.
```

Note: it's greatly recommended to execute the unit test application before each submission in the repository.

5 Usage

5.1 Encode

The encode command line is the following one:

```
$ ./build/Release/bin/encode \
    --config=./generatedConfigFiles/s3c1r3_bask/encoder.cfg \
    --frameCount=1 \
    --compressed=s3c1r3_bask.vmesh
```

5.2 Decode

The decode can be executed with:

```
./build/Release/bin/decode \
  --config=./generatedConfigFiles/s3c1r3_bask/decoder.cfg \
  --compressed=s3c1r3_bask.vmesh \
  --decMesh=s3c1r3_bask_%04d_dec.obj \
  --decTex=s3c1r3_bask_%04d_dec.png \
  --decMat=s3c1r3_bask_%04d_dec.mtl \
```

5.3 Runtime configuration and configuration files

To generate the configuration files (conmon test conditions) according to your system paths, the following action must be made:

1. copy and edit `cfg/cfg-site-default.yaml` as `cfg/cfg-site.yaml`, the paths for the binaries, sequence prefix, and the external tool configuration prefix;
2. run the ./scripts/gen-cfg.sh' script:

```
$ ./scripts/gen-cfg.sh \
    --cfgdir=./cfg/ \
    --outdir=/path/to/generated/cfgfiles
```

This operation can be executed with script './scripts/create_configuration_files.sh' and in this case the file 'cfg/cfg-site.yaml' is generated automatically according to the current folder.

```
$ ./scripts/create_configuration_files.sh

./scripts/create_configuration_files.sh Create configuration files:

    Usage:
       -o|--outdir=: configured directory                    (default: generated
```

```
   -s|--seqdir=: source sequence directory               (default:  )
   -c|--codec=:  video codec: hm, vtm                    (default: hm )
   --update:     update cfg files stored in ./cfg/vmesh/ (default: 0 )

Examples:
  ./scripts/create_configuration_files.sh
  ./scripts/create_configuration_files.sh \
    --outdir=generatedConfigFilesHM  \
    --seqdir=/path/to/contents/voxelized/ \
    --codec=hm
  ./scripts/create_configuration_files.sh \
    --outdir=generatedConfigFilesVTM \
    --seqdir=/path/to/contents/voxelized \
    --codec=vtm
  ./scripts/create_configuration_files.sh \
    --update \
    --codec=hm
```

5.4 Run experiment

An example script (`scripts/run.sh`) demonstrates how to launch the entire toolchain for a single job in the configured experiment.

This scripts starts:

- encoding process
- decoding process
- pcc metrics computation
- ibsm metrics computation

The usage of this script are presented below:

```
$ ./scripts/run.sh

Usage:
   -h|--help   : print help
   -q|--quiet  : disable logs            (default: 1 )
   -f|--frames : frame count             (default: 1 )
   -c|--cfgdir : configured directory    (default: "" )
   -o|--outdir : output directory        (default: "results" )
   --condId=   : condition: 1, 2         (default: 1 )
   --seqId=    : seq: 1,2,3,4,5,6,7,8    (default: 1 )
   --rateId=   : Rate: 1,2,3,4,5         (default: 1 )
   --tmmMetric : Use TMM metric software (default: 0 )
   --render    : Create rendered images  (default: 0 )
   --encParams : configured directory    (default: "" )
   --decParams : configured directory    (default: "" )
   --csv       : generate .csv file      (default: "" )

Examples:
  - ../scripts/run.sh
  - ../scripts/run.sh \
      --condId=1 \
      --seqId=3 \
      --rateId=3 \
      --cfgdir=generatedConfigFiles
```

```
      - ../scripts/run.sh \
          --condId=1 \
          --seqId=3 \
          --rateId=3 \
          --cfgdir=generatedConfigFiles \
          --TMMMETRIC
```

Note: The preceding script uses the `mpeg-pcc-mmetric` software and this dependency can be cloned and built with the following command line:

```
$ ./scripts/get_external_tools.sh
```

A example of execution of this script is:

```
$ ./scripts/run.sh \
    --condId=1 \
    --seqId=3 \
    --rateId=3 \
    --cfgdir=generatedConfigFiles \
    --outdir=results
Run vmesh encoder/decoder/metrics: ./scripts
Encode: results/F001/s3c1r2_bask/s3c1r2_bask
./build/Release/bin/encode \
    --config=./generatedConfigFiles/s3c1r2_bask//encoder.cfg \
    --frameCount=1 \
    --compressed=results/F001/s3c1r2_bask/s3c1r2_bask.vmesh \
   > results/F001/s3c1r2_bask/encoder.log 2>&1
Decode: results/F001/s3c1r2_bask/s3c1r2_bask
./build/Release/bin/decode \
    --config=./generatedConfigFiles/s3c1r2_bask//decoder.cfg \
    --compressed=results/F001/s3c1r2_bask/s3c1r2_bask.vmesh \
    --decMesh=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.obj \
    --decTex=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.png \
    --decMat=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.mtl \
   > results/F001/s3c1r2_bask/decoder.log 2>&1
Metrics IBSM: results/F001/s3c1r2_bask/s3c1r2_bask
./externaltools/mpeg-pcc-mmetric/build/mm \
   sequence \
    --firstFrame    1 \
    --lastFrame     1 \
   END \
   dequantize \
    --inputModel    /path/to/contents/basketball_player_fr%04d_qp12_qt12.obj \
    --outputModel   ID:deqRef \
    --useFixedPoint \
    --qp            12 \
    --minPos        "-725.812988 -483.908997 -586.02002" \
    --maxPos        "1252.02002 1411.98999 1025.34998" \
    --qt            12 \
    --minUv         "0 0" \
    --maxUv         "1.0 1.0" \
   END \
   dequantize \
    --inputModel    results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.obj \
    --outputModel   ID:deqDis \
```

```
    --useFixedPoint \
    --qp            12 \
    --minPos        "-725.812988 -483.908997 -586.02002" \
    --maxPos        "1252.02002 1411.98999 1025.34998" \
    --qt            12 \
    --minUv         "0 0" \
    --maxUv         "1.0 1.0" \
   END \
   compare \
    --mode          ibsm \
    --inputModelA   ID:deqRef \
    --inputModelB   ID:deqDis \
    --inputMapA     /path/to/contents/basketball_player_fr%04d.png \
    --inputMapB     results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.png \
    --outputCsv     results/F001/s3c1r2_bask/metric_ibsm.csv \
   > results/F001/s3c1r2_bask/metric_ibsm.log
Metrics PCC: results/F001/s3c1r2_bask/s3c1r2_bask
./externaltools/mpeg-pcc-mmetric/build/mm \
   sequence \
    --firstFrame    1 \
    --lastFrame     1 \
   END \
   dequantize \
    --inputModel    /path/to/contents/basketball_player_fr%04d_qp12_qt12.obj \
    --outputModel   ID:deqRef \
    --useFixedPoint \
    --qp            12 \
    --qt            12 \
    --minPos        "-725.812988 -483.908997 -586.02002" \
    --maxPos        "1252.02002 1411.98999 1025.34998" \
    --minUv         "0.0 0.0" \
    --maxUv         "1.0 1.0" \
   END \
   dequantize \
    --inputModel    results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.obj \
    --outputModel   ID:deqDis \
    --useFixedPoint \
    --qp            12 \
    --qt            12 \
    --minPos        "-725.812988 -483.908997 -586.02002" \
    --maxPos        "1252.02002 1411.98999 1025.34998" \
    --minUv         "0.0 0.0" \
    --maxUv         "1.0 1.0" \
   END \
   reindex \
    --inputModel    ID:deqRef \
    --sort          oriented \
    --outputModel   ID:ref_reordered \
   END \
   reindex \
    --inputModel    ID:deqDis \
    --sort          oriented \
    --outputModel   ID:dis_reordered \
```

```
   END \
   sample \
    --inputModel    ID:ref_reordered \
    --inputMap      /path/to/contents/basketball_player_fr%04d.png \
    --mode          grid \
    --useNormal \
    --useFixedPoint \
    --minPos        "-725.812988 -483.908997 -586.02002" \
    --maxPos        "1252.02002 1411.98999 1025.34998" \
    --bilinear \
    --gridSize      1024 \
    --hideProgress  1 \
    --outputModel   ID:ref_pc \
   END \
   sample \
    --inputModel    ID:dis_reordered \
    --inputMap      results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.png \
    --mode          grid \
    --useNormal \
    --useFixedPoint \
    --minPos        "-725.812988 -483.908997 -586.02002" \
    --maxPos        "1252.02002 1411.98999 1025.34998" \
    --bilinear \
    --gridSize      1024 \
    --hideProgress  1 \
    --outputModel   ID:dis_pc \
   END \
   compare \
    --mode          pcc \
    --inputModelA   ID:ref_pc \
    --inputModelB   ID:dis_pc \
    --resolution    1977.833008 \
    --outputCsv     results/F001/s3c1r2_bask/metric_pcc.csv \
   > results/F001/s3c1r2_bask/metric_pcc.log
NbOutputFaces      : 75648
TotalBitstreamBits : 150448
GridD1             : 73.833939
GridD2             : 75.434639
GridLuma           : 36.139542
GridChromaCb       : 43.351307
GridChromaCr       : 45.376358
IbsmGeom           : 46.775490
IbsmLuma           : 33.573721
EncTime            : 27.0134349
DecTime            : 0.34059222
```

The –tmmMetric paramerter executes the vmesh metric software to compute metrics rather than the mm software. In this case, the logs are as follows:

```
$ ./scripts/run.sh \
  --condId=1 \
  --seqId=3 \
  --rateId=2 \
  --cfgdir=generatedConfigFiles \
```

```
  --outdir=results \
  --tmmMetric
Encode: results/F001/s3c1r2_bask/s3c1r2_bask
./build/Release/bin/encode \
    --config=./generatedConfigFiles/s3c1r2_bask//encoder.cfg \
    --frameCount=1 \
    --compressed=results/F001/s3c1r2_bask/s3c1r2_bask.vmesh \
   > results/F001/s3c1r2_bask/encoder.log 2>&1
Decode: results/F001/s3c1r2_bask/s3c1r2_bask
./build/Release/bin/decode \
    --config=./generatedConfigFiles/s3c1r2_bask//decoder.cfg \
    --compressed=results/F001/s3c1r2_bask/s3c1r2_bask.vmesh \
    --decMesh=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.obj \
    --decTex=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.png \
    --decMat=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.mtl \
   > results/F001/s3c1r2_bask/decoder.log 2>&1
Metrics: results/F001/s3c1r2_bask/s3c1r2_bask
./build/Release/bin/metrics \
    --config=./generatedConfigFiles/s3c1r2_bask//mmetric.cfg \
    --decMesh=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.obj \
    --decTex=results/F001/s3c1r2_bask/s3c1r2_bask_%04d_dec.png \
    --frameCount=1 \
   > results/F001/s3c1r2_bask/metric_met.log
NbOutputFaces      : 75648
TotalBitstreamBits : 150448
GridD1             : 73.8339386
GridD2             : 75.434639
GridLuma           : 36.1395416
GridChromaCb       : 43.3513069
GridChromaCr       : 45.376358
IbsmGeom           : 46.7754899
IbsmLuma           : 33.5737214
EncTime            : 28.7190478
DecTime            : 0.355798779
```

5.5 Collect results

To collect the results from the log files (encoder, decoder and metric), the `./scripts/collect_results.sh` script can be uses:

```
./scripts/collect_results.sh Collect results from log files

  Usage:
    -h|--help   : print help
    -q|--quiet  : disable logs            (default: 1 )
    --condId=   : condition: 1, 2         (default: 1 )
    --seqId=    : seq: 1,2,3,4,5,6,7,8    (default: 1 )
    --rateId=   : Rate: 1,2,3,4,5         (default: 1 )
    --vdmc      : vdmc bitstream file     (default: "" )
    --logenc    : encoder log file        (default: "" )
    --logdec    : decoder log file        (default: "" )
    --logmet    : metrics log file        (default: "" )
    --csv       : generate .csv file      (default: "" )
```

```
  Examples:
    ./scripts/collect_results.sh -h
    ./scripts/collect_results.sh \
      --condId=1 \
      --seqId=3 \
      --rateId=3 \
      --vdmc=test.bin \
      --logenc=encoder.log \
      --logdec=decoder.log \
      --logmet=metric.log
```

This script can be used to parse the log files and display or get the bitrate/metric values:

```
$ ./scripts/collect_results.sh  \
  --condId  1 \
  --seqId   2 \
  --rateId  2 \
  --vdmc    s2c1r2_sold.vmesh \
  --logenc  encoder.log     \
  --logdec  decoder.log     \
  --logmet  metrics.log
2,1,2,152064,206976,72.1250986,74.0027083,29.7090586,43.1735896,43.2807054,48.36
$ RES=( $( ./scripts/collect_results.sh  \
  --condId  1 \
  --seqId   2 \
  --rateId  2 \
  --vdmc    s2c1r2_sold.vmesh \
  --logenc  encoder.log     \
  --logdec  decoder.log     \
  --logmet  metrics.log ) );
$ for((i=0;i<16;i++)); do printf "RES[%2d] = %s \n" $i ${RES[$i]}; done
RES[ 0] = 2
RES[ 1] = 1
RES[ 2] = 2
RES[ 3] = 152064
RES[ 4] = 206976
RES[ 5] = 72.1250986
RES[ 6] = 74.0027083
RES[ 7] = 29.7090586
RES[ 8] = 43.1735896
RES[ 9] = 43.2807054
RES[10] = 48.3653428
RES[11] = 29.1616112
RES[12] = 24.8878219
RES[13] = 0.257652824
RES[14] = 484.457
RES[15] = 178.211
```

5.6 Run all experiments and create render and graph pdf files

To run CTC experiments with all sequences, all conditions and all rates as definied in CTC conditions, the `./scripts/run_all.sh`script can be used :

```
$ ./scripts/run_all.sh --help
./scripts/run_all.sh execute all encoding/decoding/metrics
```

```
  Usage:
    -h|--help      : print help
    -q|--quiet     : disable logs                    (default: 1 )
    -f|--frames    : frame count                     (default: 1 )
    -c|--cfgdir    : configured directory            (default: "generatedConfigF
    -o|--outdir    : output directory                (default: experiments )
    --experiments  : csv configuration files         (default: /srv/wp21/PCC/ric
    --tmmMetric    : Use TMM metric software         (default: 0 )
    -t|--threads   : Number of parallel experiments  (default:  )
    --render       : Create pdf  rendered images     (default: 0 )
    --graph        : Create pdf with metric graphs   (default: 0 )
    --xlsm         : Create CTC xlsm files           (default: 0 )

  Examples:
    ./scripts/run_all.sh -h
    ./scripts/run_all.sh \
      --experiments ./scripts/test.csv \
      --outdir      experiments \
      --cfgdir      generatedConfigFilesHM \
      --frame       2 \
      --graph \
      --render \
      --xlsm \
      --quiet
```

This scripts executes severals experiments that must be defined in `./scripts/test.csv` files. This file defined the experiments that must be evaluated, one experiments by line. Each experiments must set:

- Name: the name of the experiment.
- EncParams: the encoder parameters used.
- DecParams: the decoder parameters used.

An example of this file is the following one:

```
$ cat ./scripts/test.csv
Name,EncParams,DecParams
anchor,,
texture1k,--textureVideoWidth=1024 --textureVideoHeight=1024,
texture2k,--textureVideoWidth=2048 --textureVideoHeight=2048,
```

The experiments can be executed with the following command line:

```
$ ./scripts/run_all.sh \
  --frame=4 \
  --threads 10 \
  --render \
  --graph \
  --xlsm \
  --quiet
```

The `--graph` and `--renderer` options create pdf files with the graph and the render images of all the experiments defined in `./scripts/test.csv`. Examples of the created pdf files can be seen in figures 2 and 3.

The `--xlsm` option fill the CTC XLSM spreadsheet with the results of the current experiences. The first line of the CSV file is set as anchor of the experiences and the other one are compared to the anchor and between them. With the previously presented CSV files teh following files are created:

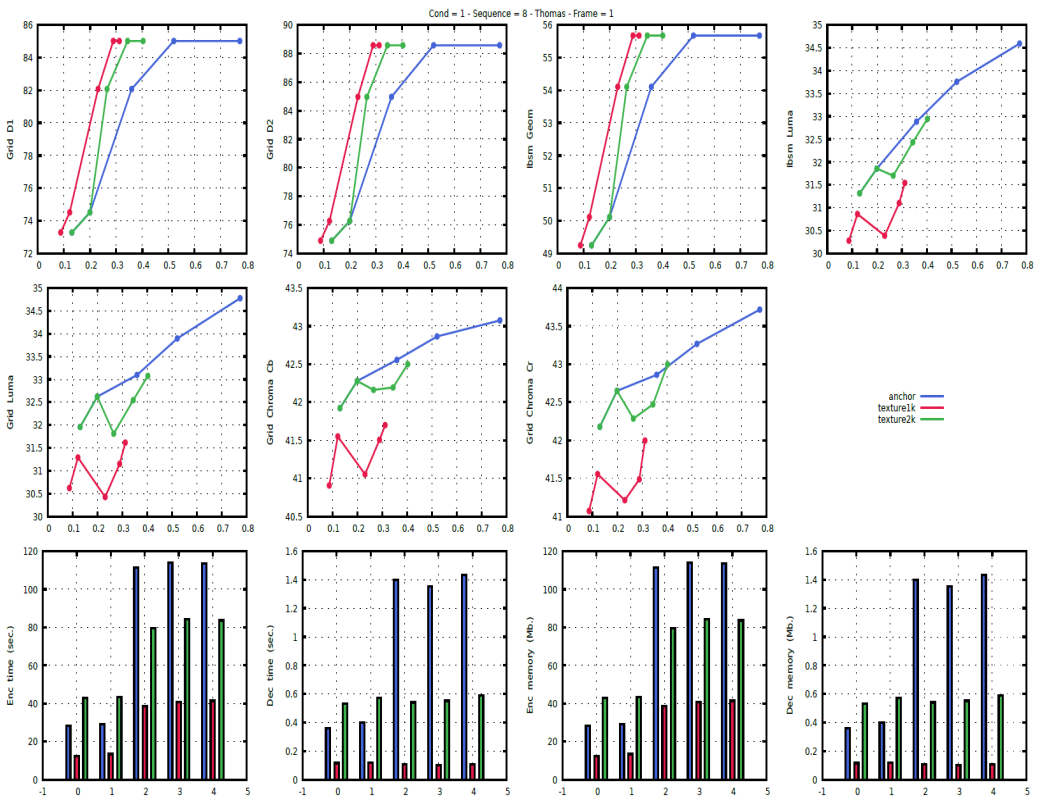

Figure 2 – Example of graphs.

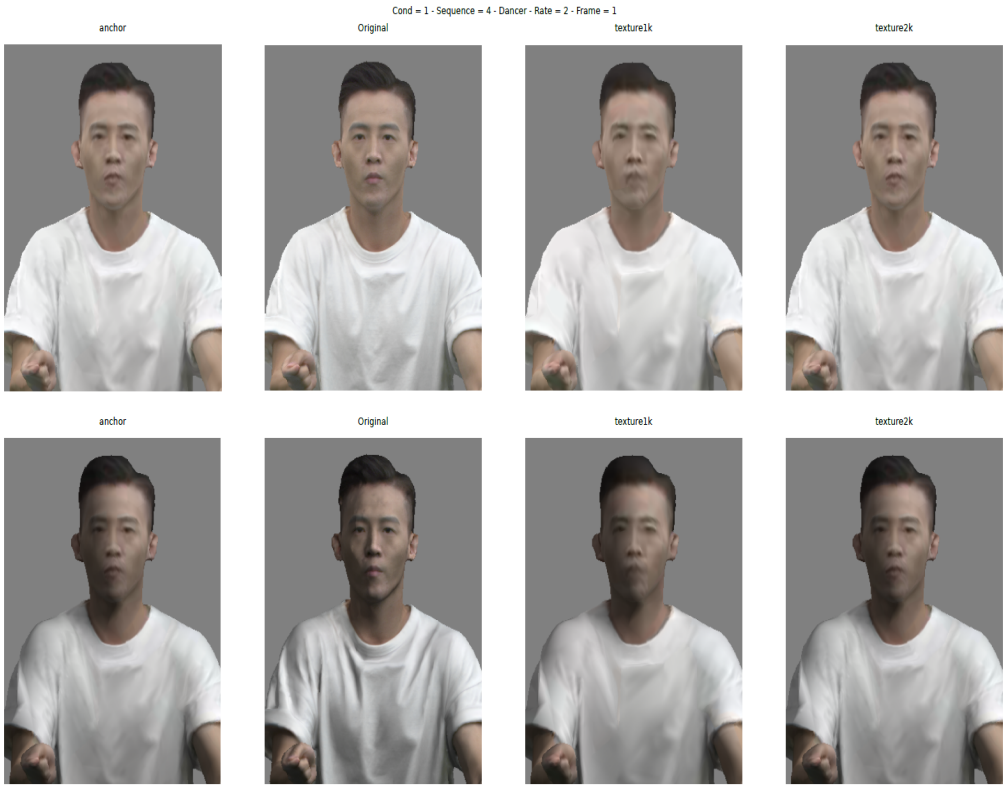

Figure 3 – Example of render images.

- ./experiments/F004_anchor_vs_texture1k.xlsm
- ./experiments/F004_anchor_vs_texture2k.xlsm
- ./experiments/F004_texture1k_vs_texture2k.xlsm

Note: The `--xlsm` option uses `openpyxl` Python module to fill the XLSM files and requieres Python3 to work properly.

The `--threads N` option allows experiments to be run in parallel with N which defines the number of parallel tests.

Note: This option has been used on Linux and uses Linux commands to work. Please, use this script in a Linux terminal. On Window, please uses: msys, cygwin, mingw or Windows Subsystem for Linux (WSL),

6 Main software input parameters

The following subsections contain input parameters for encoding, decoding, and metrics software.

6.1 Encode software input parameters

–key=value	**Usage**
Common	
–help=0	This help text
-c, –config=...	–config=... Configuration file name
-v, –verbose=1	–verbose=1 Verbose output
Input	
–srcMesh=""	Input mesh
–srcTex=""	Input texture
–positionBitDepth=12	Input positions bit depth
–texCoordBitDepth=12	Input texture coordinates bit depth
–startFrameIndex=0	First frame number
–frameCount=1	Number of frames
–framerate=30	Frame rate
Output	
–compressed=""	Compressed bitstream
–recMesh=""	Reconstructed mesh
–recTex=""	Reconstructed texture
–dequantizeUV=1	Dequantize texture coordinates of the reconstructed meshes
–recMat=""	Reconstructed materials
General	
–keep=0	Keep intermediate files
–checksum=1	Compute checksum
Group of frames analysis	
–gofMaxSize=32	Maximum group of frames size
–analyzeGof=0	Analyze group of frames
Geometry decimate	
–target=0.125	Target triangle count ratio

–key=value	Usage
–minCCTriangleCount=0	minimum triangle count per connected component
–minPosition="0,0,0"	Min position
–maxPosition="0,0,0"	Max position
Texture parametrization	
–textureParametrizationQuality=DEFAULT	Quality level of DEFAULT, FAST or QUALITY
–textureParametrizationMaxCharts=0	Maximum number of charts to generate
–textureParametrizationMaxStretch=0.16667	Maximum amount of stretch 0 to 1
–textureParametrizationGutter=2	Gutter width betwen charts in texels
–textureParametrizationWidth=512	texture width
–textureParametrizationHeight=512	texture height
Geometry parametrization	
–baseIsSrc=0	Base models are src models
–subdivIsBase=0	Subdiv models are src models
–subdivInter=0	Subdiv inter
–subdivInterWithMapping=0	Subdiv inter with mapping
–maxAllowedD2PSNRLoss=1	Maximum allowed D2 PSNR Loss
Intra geometry parametrization	
–ai_sdeform=1	Apply deformation refinement stage
–ai_subdivIt=3	Subdivision iteration count
–ai_forceNormalDisp=0	Force displacements to aligned with the surface normals
–ai_unifyVertices=1	Unify duplicated vertices
–ai_deformNNCount=1	Number of nearest neighbours used during the initial deformation stage
–ai_deformNormalThres=0.1	Maximum allowed normal deviation during the initial deformation stage
–ai_sampIt=3	Number of subdivision iterations used for geometry sampling
–ai_fitIt=16	Number of iterations used during the deformation refinement stage
–ai_smoothCoeff=0.25	Initial smoothing coefficient used to smooth the deformed mesh during deformation refinement
–ai_smoothDecay=0.75	Decay factor applied to intial smoothing coefficient after every iteration of deformation refinement
–ai_smoothMissedCoeff=0.1	Smoothing coefficient applied to the missed vertices
–ai_smoothMissedIt=10	Number of iterations when smoothing the positions of the missed vertices
–ai_smoothMethod=1	Smoothing method to be applied when smoothing the deformed mesh duringthe deformation refinement stage
–ai_deformUpdateNormals=1	Recompute normals after each iteration of deformation refinement
–ai_deformFlipThres=-0.5	Threshold to detect triangle normals flip
–ai_useInitialGeom=1	Use the initial geometry during the the deformation refinement stage

–key=value	Usage
–ai_fitSubdiv=1	Update the positions of the decimated mesh to minimize displacements between the subdivided mesh and the deformed mesh
–ai_smoothMotion=1	Apply smoothing to motion instead of vertex positions
Inter geometry parametrization	
–ld_sdeform=1	Apply deformation refinement stage
–ld_subdivIt=3	Subdivision iteration count
–ld_forceNormalDisp=0	Force displacements to aligned with the surface normals
–ld_unifyVertices=1	Unify duplicated vertices
–ld_deformNNCount=1	Number of nearest neighbours used during the initial deformation stage
–ld_deformNormalThres=0.1	Maximum allowed normal deviation during the initial deformation stage
–ld_sampIt=3	Number of subdivision iterations used for geometry sampling
–ld_fitIt=16	Number of iterations used during the deformation refinement stage
–ld_smoothCoeff=0.25	Initial smoothing coefficient used to smooth the deformed mesh during deformation refinement
–ld_smoothDecay=0.75	Decay factor applied to intial smoothing coefficient after every iteration of deformation refinement
–ld_smoothMissedCoeff=0.1	Smoothing coefficient applied to the missed vertices
–ld_smoothMissedIt=10	Number of iterations when smoothing the positions of the missed vertices
–ld_smoothMethod=1	Smoothing method to be applied when smoothing the deformed mesh duringthe deformation refinement stage
–ld_deformUpdateNormals=1	Recompute normals after each iteration of deformation refinement
–ld_deformFlipThres=-0.5	Threshold to detect triangle normals flip
–ld_useInitialGeom=1	Use the initial geometry during the the deformation refinement stage
–ld_fitSubdiv=1	Update the positions of the decimated mesh to minimize displacements between the subdivided mesh and the deformed mesh
–ld_smoothMotion=1	Apply smoothing to motion instead of vertex positions
Lifting	
–liftingIterationCount=2	Lifting subdivision iteration count
–liftingQP="16,28,28"	Quantization parameter for displacements
–liftingBias="0.333333,0.333333,0.333333"	Quantization bias for displacements
Base mesh	
–baseMeshPositionBitDepth=10	Quantization bits for base mesh positions
–baseMeshTexCoordBitDepth=8	Quantization bits for base mesh texture coordinates

–key=value	Usage
–invertOrientation=0	Invert triangles orientation
–unifyVertices=0	Unify duplicated vertices
–meshCodecId=255	Mesh codec id
Geometry video	
–encodeGeometryVideo=1	Encode displacements video
–geometryVideoCodecId=HM	Geometry video codec id
–geometryVideoEncoderConfig=“”	Geometry video config file
Texture video	
–encodeTextureVideo=1	Encode texture video
–textureVideoCodecId=HM	Texture video codec id
–textureVideoEncoderConfig=“”	Texture video encoder configuration file
–textureVideoEncoderConvertConfig=“”	HDRTools encode configuration file
–textureVideoDecoderConvertConfig=“”	HDRTools decode configuration file
–textureVideoDownsampleFilter=4	Chroma downsample filter in [0;22]
–textureVideoUpsampleFilter=0	Chroma upsample filter in [0;7]
–textureVideoFullRange=0	Texture video range: 0: limited, 1: full
–textureVideoQP=8	Quantization parameter for texture video
–textureVideoWidth=2048	Output texture width
–textureVideoHeight=2048	Output texture height
Metrics	
–pcc=0	Compute pcc metrics
–ibsm=0	Compute ibsm metrics
–pcqm=0	Compute pcqm metrics
–gridSize=1024	Grid size
–resolution=0	Resolution
–pcqmRadiusCurvature=0.001	PCQM radius curvature
–pcqmThresholdKnnSearch=20	PCQM threshold Knn search
–pcqmRadiusFactor=2	PCQM radius factor
Caching	
–cachingDirectory=“”	Caching directory
–cachingPoint=none	Caching points: - 0/none : off - 1/simplify: symplify - 1/uvatlas : symplify - 2/subdiv : subdiv - 255/create: create caching files

6.2 Decode software input parameters

–key=value	Usage
Common	
–help=0	This help text
-c, –config=...	–config=... Configuration file name
-v, –verbose=1	–verbose=1 Verbose output
Input	
–compressed=“”	Compressed bitstream

–key=value	Usage
Output	
–decMesh=“”	Decoded mesh
–decTex=“”	Decoded texture
–decMat=“”	Decoded materials
–dequantizeUV=1	Dequantize texture coordinates of the decoded meshes
–startFrameIndex=0	First frame number
–framerate=30	Frame rate
General	
–keep=0	Keep intermediate files
–checksum=1	Compute checksum
Decoder	
–textureVideoDecoderConvertConfig=“”	HDRTools decode cfg
–textureVideoUpsampleFilter=0	Chroma upsample filter in [0;7]
–textureVideoFullRange=0	Texture video range
Metrics	
–pcc=0	Compute pcc metrics
–ibsm=0	Compute ibsm metrics
–pcqm=0	Compute pcqm metrics
–gridSize=1024	Grid size
–minPosition=“0,0,0”	Min position
–maxPosition=“0,0,0”	Max position
–positionBitDepth=12	Position bit depth
–texCoordBitDepth=13	Texture coordinate bit depth
–pcqmRadiusCurvature=0.001	PCQM radius curvature
–pcqmThresholdKnnSearch=20	PCQM threshold Knn search
–pcqmRadiusFactor=2	PCQM radius factor
–resolution=0	resolution
–startFrameIndex=0	Metric frame start
–frameCount=0	Metric frame count
–srcMesh=“”	Metric Source mesh path
–srcTex=“”	Source texture path

6.3 Metrics software input parameters

–key=value	Usage
–help=0	This help text
-c, –config=...	–config=... Configuration file name
-v, –verbose=0	–verbose=0 Verbose output
Source	
–srcMesh=“”	Source mesh
–srcTex=“”	Source texture
Decoded	
–decMesh=“”	Reconsctructed/decoded mesh
–decTex=“”	Reconsctructed/decoded texture

–key=value	Usage
Sequence	
–startFrameIndex=1	First frame number
–frameCount=1	Number of frames
–minPosition="0,0,0"	Min position
–maxPosition="0,0,0"	Max position
–positionBitDepth=12	Position bit depth
–texCoordBitDepth=13	Texture coordinate bit depth
–dequantizeUV=1	Texture coordinates of the decoded meshes are quantized
PCC metric	
–pcc=0	Compute pcc metrics
–gridSize=1024	Grid size
–resolution=0	Resolution
IBSM metric	
–ibsm=0	Compute ibsm metrics
PCQM metric	
–pcqm=0	Compute PCQM metrics
–pcqmRadiusCurvature=0.001	PCQM radius curvature
–pcqmThresholdKnnSearch=20	PCQM threshold Knn search
–pcqmRadiusFactor=2	PCQM radius factor

7 Licence

The copyright in this software is being made available under the BSD Licence, included below. This software may be subject to other third party and contributor rights, including patent rights, and no such rights are granted under this licence.

Copyright (c) 2022, ISO/IEC
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of the ISO/IEC nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE

LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8 Contacts and reporting issues

For any issues or questions don't hesitate to open issues in V-Mesh git repository or to contact us:

- Julien Ricard (julien.ricard@interdigital.com)
- Wenjie Zou (wjzou@xidian.edu.cn)

Bugs should be reported on the issue tracker at http://mpegx.int-evry.fr/software/MPEG/vmesh/TM/mpeg-vmesh-tmc/issues.