Companion Paper for "MiniView Layout for Bandwidth-Efficient 360-Degree Video"

Mengbai Xiao
The Ohio State University
xiao.736@osu.edu

Shuoqian Wang
SUNY Binghamton
swang130@binghamton.edu

Chao Zhou
SUNY Binghamton
czhou5@binghamton.edu

Li Liu
George Mason University
lliu8@gmu.edu

Zhenhua Li
Tsinghua University
lizhenhua1983@tsinghua.edu.cn

Yao Liu
SUNY Binghamton
yaoliu@binghamton.edu

Songqing Chen
George Mason University
sqchen@gmu.edu

Lucile Sassatelli
Universite Cote d'Azur, CNRS, I3S
sassatelli@i3s.unice.fr

Gwendal Simon
IMT Atlantique
gwendal.simon@imt-atlantique.fr

ABSTRACT

This artifact includes source code, scripts and datasets required to reproduce the experimental figures in the evaluation of the MM'18 paper, which is entitled "MiniView Layout for Bandwidth-Efficient 360-Degree Video" [3]. The artifact reports the comparison results among the standard cube layout (CUBE), the equi-angular layout (EAC), and the MiniView layout (MVL) in terms of compressed video size, visual quality of views and decoding and rendering time.

CCS CONCEPTS

• **Information systems → Multimedia content creation**.

KEYWORDS

360 video, virtual reality, view-adaptive encoding

ACM Reference Format:
Mengbai Xiao, Shuoqian Wang, Chao Zhou, Li Liu, Zhenhua Li, Yao Liu, Songqing Chen, Lucile Sassatelli, and Gwendal Simon. 2019. Companion Paper for "MiniView Layout for Bandwidth-Efficient 360-Degree Video". In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3343031.3351168

1 ARTIFACT DESCRIPTION

The artifact is available at

`https://github.com/bingsyslab/mm19-artifact`

without the datasets used in the evaluation. The datasets are accessible at

`https://dl.dropboxusercontent.com/s/snlomfjoh7ybsk3/dataset.tgz`

, and it can also be automatically downloaded by running the script in the artifact. The artifact includes a patch to `ffmpeg` (`360-project` filter), running scripts, documentations and a small dataset for verification.

MM '19, October 21–25, 2019, Nice, France

ACM ISBN 978-1-4503-6889-6/19/10…$15.00
https://doi.org/10.1145/3343031.3351168

1.1 Folder structure

The folder structure of the artifact is shown as follows:

```
mm19-artifact/
|- makefile
|- makefile.sub
|- cdf.pl
|- cdf.gp
|- config-output-pdf.gp
|- remap.pl
|- README.md
|- 360_project.patch
|- layouts/
   |- %.lt
|- shaders/
   |- %.glsl
|- Diving/
```

`makefile`: the script controlling experimental setups, evaluations, and plotting.
`makefile.sub`: the script carrying out evaluations for one video.
`cdf.pl`: the script collecting metrics from a set of videos.
`config-output-pdf.gp`, `cdf.gp`: the scripts plotting CDF figures.
`remap.pl`: the script producing 360-degree videos with a specific layout from an HD equirectangular video.
`README.md`: the documentation of the artifact.
`360_project.patch`: the patch file to `ffmpeg` that installs the `360-project` filter.
`layouts/`: the folder holding different layout files.
`shaders/`: the folder holding relevant OpenGL shaders.
`Diving/`: the example video dataset.

1.2 Datasets

The videos and user traces are from publicly available datasets [1, 2]. For fairness, we transcode all videos into the equirectangular layout at the resolution of 3840x2048, and these videos are chunked into segments of 1 second. In addition, the user orientation traces are unified into `.txt` files, in which a line is a sample and it looks like

```
0.141033 4.000000 -15.787280 174.084166
```

. The items are the timestamp in seconds, the line number, and the head rotations surrounding x-axis and y-axis, respectively.

Table 1: Dataset description

	# of traces	Video names
Dataset-1 [1]	53	**Diving**, Paris, **Rollercoaster**, Timelapse, Venise
Dataset-2 [2]	48	Conan_Gore_Fly, Cooking_Battle, **Front**, Help, Rhinos, Conan_Weird_Al, Football, **Tahiti_Surf**, FemaleBasketball, Fighting, Korean, Reloaded, RioVR, TFBoy, VoiceToy, Anitta

We ignore the head rotation surrounding z-axis since users rarely change their head around this axis. The video segments and the orientation files are named in the formats of `${sec}_sec.mp4` and `uid-${uid}_raw.txt`, respectively.

The datasets include 21 videos. 5 of them have 58 users' view orientations while the other 16 videos have 48 users' head movement traces. The list of these video names are shown in Table 1. In the evaluation, we distinguish them into two groups: *moving-camera* and *static-camera* videos. In Table 1, the *moving-camera* videos are presented in bold.

2 INSTALLATION

Before the experiment workflow, an `ffmpeg` with `360-project` filter need to be built and all datasets should be downloaded.

2.1 Dependencies

The `360-project` filter has been tested with the `master` branch of `ffmpeg` with the ref

`37e4c226c06c4ac6b8e3a0ccb2c0933397d6f96f`

and is expected to be correctly built with the latest `ffmpeg` source code. The experiments are successfully carried out on Ubuntu 18.04 and Ubuntu 16.04. The tested OpenGL version is OpenGL 3.3 (Core Profile) Mesa 18.0.5. The scripts in the artifact use Gnuplot 5.2 patchlevel 2, Perl 5.018 and GNU Make 4.1.

To install the software dependencies on a Ubuntu system, run the following commands:

```
$ sudo apt-get install autoconf automake build-essential \
  cmake git-core libass-dev libfreetype6-dev libsdl2-dev \
  libtool libva-dev libvdpau-dev libvorbis-dev libxcb1-dev \
  libxcb-shm0-dev libxcb-xfixes0-dev pkg-config \
  texinfo wget zlib1g-dev libglew-dev libglfw3-dev \
  libx264-dev gnuplot make yasm
```

Hardware dependencies: Running this artifact on a platform equipped with a GPU is highly recommended, which can greatly accelerate the projection operations.

2.2 Running artifact over `ssh`

Since the artifact is developed based on GLFW, it requires a display to create a window, though it is invisible. For running this artifact on a server without a monitor connected via `ssh`, the virtual framebuffer needs to be installed:

```
$ sudo apt-get install xvfb
```

In the experiment workflow, two commands ("`make remap-videos`" and "`make view-videos`") processing videos with OpenGL need to be prefixed with `svfb-run` to ensure their successful run. For example:

```
$ xvfb-run make view-videos VNAMES=Diving
```

Details of these two commands are discussed in Section 3.1.

2.3 Artifact installation

(1) Download the artifact with the following instructions:

```
$ git clone https://github.com/bingsyslab/mm19-artifact
```

(2) Download, patch, and build `ffmpeg` with `360-project` filter.

- Use the `makefile` script to automatically download, patch, and build `ffmpeg`:

```
$ cd mm19-artifact
$ make prepare-ffmpeg
```

If `ffmpeg` has been successfully built, two binaries `ffmpeg/ffmpeg` and `ffmpeg/ffprobe` can be found.

- If manually building `ffmpeg` is preferred, follow the instructions below:

```
$ cd mm19-artifact
$ git clone https://github.com/FFmpeg/FFmpeg ffmpeg
$ cd ffmpeg
$ git apply ../360_project.patch
$ ./configure --enable-gpl --enable-libx264 \
    --extra-libs='-lglfw -lGLEW -lGL -lGLU'
$ make -j8 ffmpeg ffprobe
```

(3) To download the datasets:

```
$ make prepare-dataset
```

This will automatically download a tarball of the dataset named `dataset.tgz` and extract it into a list of folders. The tarball is 6.9G and the dataset takes additional 7.6G disk space after decompression. The tarball can also be accessed via the link shown in Section 1. A video to be tested is organized in the format:

```
Diving/
 |- 1280/
    |- %.mp4
    |- %.txt
```

For a video "Diving", it should contain the HD equirectangular 360-degree video segments and the orientation files. Both of them are stored in the directory `1280/`. Following the details in Section 1.2 helps add a new test video into the experiment.

3 EXPERIMENT WORKFLOW

The experiment follows three major steps: 1) 360-degree video generation, 2) view video generation, and 3) experimental measurements. A full run of this artifact on the provided 21 videos (with the default configurations) takes ~12 hours, and an additional ~80G disk space will be occupied. The workflow can also be carried out on a single video for proving its completeness. In this section, we

Table 2: Experimental parameters that can be customized

Name	Description	Value format	Default value	Value as a vector?
VNAMES	The folder names of tested videos	{ Diving, Paris, ...}	All videos	Yes
MOVING_VNAMES	The names of videos as moving-camera type	{ Diving, Paris, ...}	Diving, Rollercoaster, Tahiti_Surf, Front	Yes
LAYOUTS	The names of tested layouts	{cube, eac, mv}	cube eac mv	Yes
SCHEMES	The compression parameters passed to x264	{crf${N}}	crf23	Yes
UID_NR	The number of users' orientation traces to test	${N}	20	No
TIMES_NR	The number of video segments to test	${N}	10	No
CUBE_RES	The resolution of CUBE layout	${RES}x${RES}	1920x1280	No
EAC_RES	The resolution of EAC layout	${RES}x${RES}	1920x1280	No
MV_RES	The resolution of MiniView layout	${RES}x${RES}	2240x832	No
FOV	The FoV of generated view videos	${DEGREE}x${DEGREE}	100x100	No
VIEW_RES	The resolution of generated view videos	${RES}x${RES}	800x800	No

will first present how to carry out the experiment on an example video "Diving". Then we have an integrated command to run the full experiment presented in our original paper. We will further show how to customize this artifact.

3.1 Video generation

(1) To generate 360-degree videos for a given video:

```
$ make remap-videos VNAMES=Diving
```

This command generates different layouts of 360-degree videos, where the option VNAMES specifies which video is used. This artifact is extensively customizable and more details can be found in Section 3.4. The output videos are generated into crf23/remaps/ in the names of ${layout}_${time}.mp4.

(2) To generate view videos:

```
$ make view-videos VNAMES=Diving
```

The view videos directly generated from the HD equirectangular videos (1280/${time}.mp4) are named as %.hd.mp4 in views/ while the view videos generated from different layouts of 360-degree videos are located at crf23/views/.

3.2 Video analysis

(3) To measure various metrics:

```
$ make psnr-logs VNAMES=Diving
$ make ssim-logs VNAMES=Diving
$ make ts-logs VNAMES=Diving
```

These commands measure PSNR, SSIM as well as rendering and decoding time, which are preserved as log files stored at crf23/psnr/, crf23/ssim/, and crf23/ts/, respectively. With the videos and log files generated, a test video folder looks like:

```
|- Diving/
  |- 1280/
    |- ${time}.mp4
    |- uid-${uid}_raw.txt
  |- crf23/
    |- remaps/
      |- ${layout}_${time}.mp4
      |- views/
        |- ${time}.${uid}.${layout}.mp4
      |- psnr/
        |- ${time}.${uid}.${layout}.log
      |- ssim/
        |- ${time}.${uid}.${layout}.log
      |- ts/
        |- ${time}.${uid}.${layout}.log
    |- views/
      |- ${time}.${uid}.hd.mp4
```

(4) Generate the figures in the PDF format:

```
$ make cdfs pdfs VNAMES=Diving
```

Two types of files are generated with this command. %.cdf is the data file and %.pdf files are the figures. There will be four figures showing the measured PSNR (psnr_avg.crf23-psnr.pdf), SSIM (All.crf23-ssim.pdf), video size (size.crf23-remaps.pdf) and rendering time (ts.crf23-ts.pdf).

3.3 Re-run the full experiment

To repeat the experiments presented in our original paper, use the following command

```
$ make psnr-ssim-ts-rand moving-static-cdfs moving-static-pdfs
```

The configuration of this experiment is the default ones shown in Section 3.4. It worth noting that this will generate 7 figures since we distinguish the 21 videos into *moving-camera* videos and *static-camera* videos for PSNR, SSIM and video size measurement.

3.4 Experiment customization

The experiment can be extensively customized. The comprehensive customizable parameters are listed in Table 2. An example of customized experiment can be launched like

```
$ make psnr-ssim-ts-rand VNAMES=Paris \
  FOV=90x90 LAYOUTS=mv SCHEMES=crf18 UID_NR=1 TIMES_NR=5
```

This means we only carry experiments over the Paris video and only test the MiniView layout. When generating the MiniView videos, the compression parameter uses crf=18. The FoV of view

videos is 90°x90°. We randomly test over 5 video segments and 1 orientation trace.

4 `360-PROJECT` FILTER

We implement `360-project` as an `ffmpeg` filter that generates a user view from an input 360-degree frame. Understanding this tool is not necessary to evaluate the MiniView Layout. We will give a brief description to this tool and further details can be found in the documentation of this artifact.

An example of using this filter is like:

```
$ ./ffmpeg -loglevel 'info' -y -i cube.mp4 -filter:v \
  "project=800:800:90:90:0:180:0:vertex.glsl:eqdis.glsl::cube.lt:0.0:1.0" \
  back.mp4
```

This command generates a view video from the 360-degree video `cube.mp4`. The output view video has the resolution of 800x800, and its FoV is 90°x90°. The video shows what a user will watch as she turns her head following {0°, 180°, 0°}, which is exactly the view from her back. An orientation file can be passed in to simulate the user's head movement. `vertex.glsl` and `eqdis.glsl` are the vertex shader and the fragment shader used to produce the view video. `cube.lt` is the layout file describing the input video.

5 MISCELLANEOUS REMARKS

- In the artifact, a list of commands can be used to ease clean operations:

```
$ make clean-remaps
$ make clean-views
$ make clean-logs
$ make clean-cdfs
$ make clean-pdfs
```

These commands remove remapped videos, view videos, experimental logs, CDF data files, and PDF figures, respectively.

- If you prefer building the `ffmpeg` at another place, you need to redirect the `ffmpeg` folder in the `makefile`:

```
export FFMPEG_DIR := ${PATH_TO_FFMPEG}
```

6 NOTES FROM REVIEWERS

Two co-authors of this paper are reviewers of this companion paper. We have successfully installed the library and reproduced the results that were initially presented in the original paper [3]. We also acknowledge the efforts of the authors of the original article to make the reproducibility process as smooth as possible. We would like in particular to emphasize two key improvements that have been implemented during the reviewing process to make the code source easier to manipulate for other researchers.

First, the initial code was presented such as it was tempting to manipulate the whole package as a "black box." One unique command was supposed to launch the full process of preparing the video, testing the quality, collecting the data, and plotting the figures. It was still possible to dive into the code source if one wanted to make the process step by step, but the main *make* commands did not allow a fraction of the whole processing. We believe that other researchers interested in testing the new layouts are interested by implementing either only the video processing part (typically to build the miniview videos and to run other types of evaluation on this layout), or by implementing only the evaluation part (typically to test the results of their own projection against miniview layout in the same performance evaluation configuration). As a result of this observation, and after a redefinition of the script commands, the original unique command has been split into multiple commands, which provide more flexibility for researchers in the area.

Second, we were initially not able to run the code properly due to the `ssh` issue explained in Section 2.2. We were at this time puzzled: on the one hand our mission was essentially to reproduce the results, in the sense that our primary goal was to check whether the scientific results are true. Even though it is more comfortable to set a machine with the right configuration from remote public data-centers, this comfort is not a requirement to assess that the results are valid. On the other hand, the code that is presented in the paper is supposed to run in a server, potentially many servers in a content delivery system. These machines do not have a display monitor and can only be accessed remotely. It is our understanding that one of the motivations for sharing code is to ease the development of prototypes based on the proposal by third-party researchers. It was clear that this objective cannot be fulfilled without `ssh`-friendly implementation. Fortunately the original authors proposed some fixes, which not only allow us to run the code as in the original paper in a comfortable way, but also allow third-party researchers to implement part of this code in a prototype.

Overall we have been able to reproduce the initial experiments with similar results as what was presented in the original paper. The code now enables running a fraction of the code, hopefully to the benefit of other researchers, and it can run in any data-center, with respect to the hardware and software requirements.

ACKNOWLEDGMENTS

This work is partially supported by NSF under grants CNS-1524462 and CNS-1618931.

REFERENCES

[1] Xavier Corbillon, Francesca De Simone, and Gwendal Simon. 2017. 360-Degree Video Head Movement Dataset. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. ACM, 199–204.

[2] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. 2017. A Dataset for Exploring User Behaviors in VR Spherical Video Streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. ACM, 193–198.

[3] Mengbai Xiao, Shuoqian Wang, Chao Zhou, Li Liu, Zhenhua Li, Yao Liu, and Songqing Chen. 2018. MiniView Layout for Bandwidth-Efficient 360-Degree Video. In *Proceedings of the 26th ACM International Conference on Multimedia (MM '18)*. ACM, 914–922.