LF Coding Framework description (Release version 1.2)

==

This document describes the Light Field (LF) Coding Framework proposed by Ahmad *et al*. in under-review paper, "Computationally efficient light field image compression using a multiview HEVC framework", IEEE Access September 2019. The coding framework reads the input LF image, generates multiview sequences, encodes the generated sequences using MVHEVC encoder and finally produces the decoded Mat files.

LF Framework MATLAB Code usage instructions

===

Prerequisites

The following tool and datasets are required to run the code:

- Matlab
- JPEG Pleno LF Datasets

The LF datasets are available at the following links:

LF Name	Source	Link
Bikes	Lytro	https://jpeg.org/plenodb/lf/pleno_lf/Bikes.zip
Danger de Mort	Lytro	https://jpeg.org/plenodb/lf/pleno_lf/Danger_de_Mort.zip
Fountain Vincent 2	Lytro	https://jpeg.org/plenodb/lf/pleno_lf/Fountain_Vincent2.zip
Stone Pillars Outside	Lytro	https://jpeg.org/plenodb/lf/pleno_lf/Stone_Pillars_Outside.zip
Set2 2K sub	Fraunhofer IIS	https://jpeg.org/plenodb/lf/pleno_lf/set2.zip
Tarot Cards	Stanford	http://lightfield.stanford.edu/data/tarot_fine_lf/rectified.zip

Instructions

Step 1. The first step is to download the LF datasets from the above links and put them in the "/Datasets/[Source]/" folder as shown in Fig. 1, where the "Source" is either Lytro, Fraunhofer_IIS, or Stanford:

Note: Rename the downloaded Stanford Tarot Cards LF from "rectified" to "tarot_Stanford".

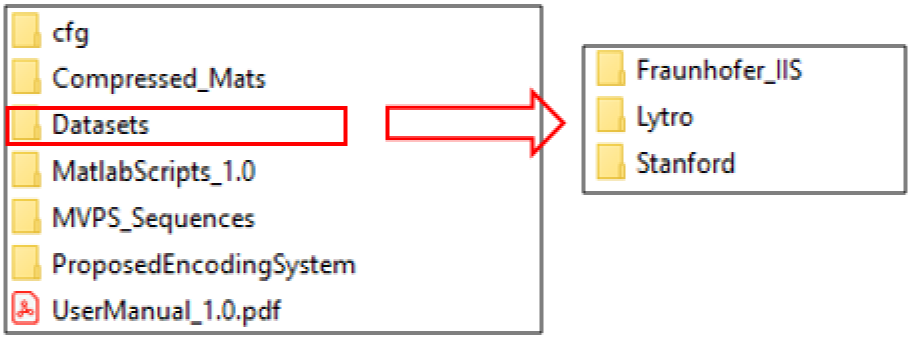
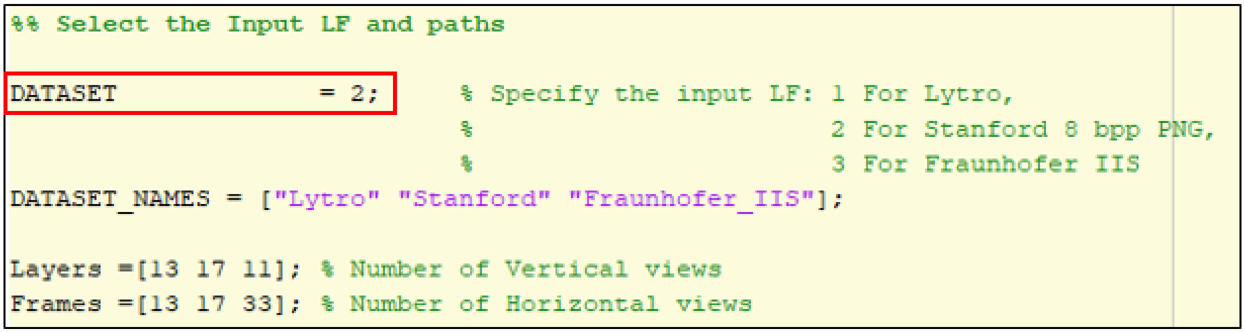


Fig. 1: LF dataset placement hierarchy.

Step 2. Next, open the "/MatlabScripts_1.2/MIUN_LFCF.m" file in MATLAB and set the "Dataset" parameter according to desired input LF.



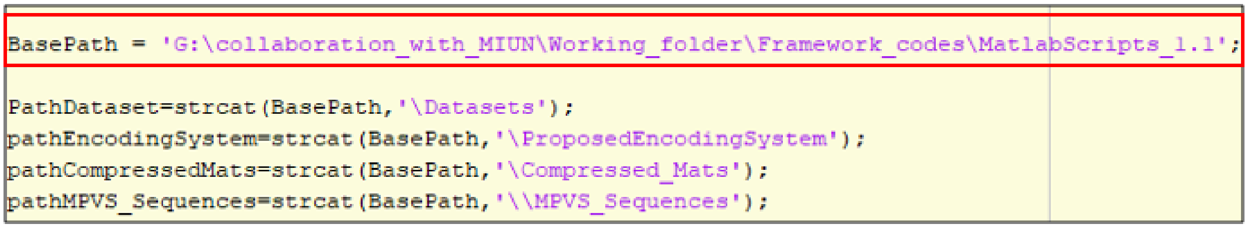
```
%% Select the Input LF and paths

DATASET              = 2;      % Specify the input LF: 1 For Lytro,
                               %                       2 For Stanford 8 bpp PNG,
                               %                       3 For Fraunhofer IIS
DATASET_NAMES = ["Lytro" "Stanford" "Fraunhofer_IIS"];

Layers =[13 17 11]; % Number of Vertical views
Frames =[13 17 33]; % Number of Horizontal views
```

Fig. 2: Select input LF dataset.

Also, provide the base folder path of the MIUN_LFCF framework folder as indicated by red box in Fig. 3. **Please ensure that the full path does not contain any empty spaces.**

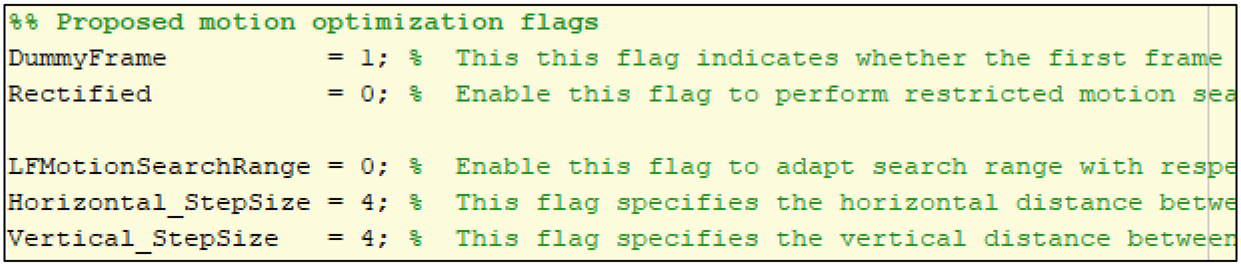


```
BasePath = 'G:\collaboration_with_MIUN\Working_folder\Framework_codes\MatlabScripts_1.1';

PathDataset=strcat(BasePath,'\Datasets');
pathEncodingSystem=strcat(BasePath,'\ProposedEncodingSystem');
pathCompressedMats=strcat(BasePath,'\Compressed_Mats');
pathMPVS_Sequences=strcat(BasePath,'\\MPVS_Sequences');
```

Fig. 3: Specify the full folder path.

Step 3. Now enable/disable the proposed motion optimization using the flags shown in Fig. 4.



```
%% Proposed motion optimization flags
DummyFrame          = 1; %  This this flag indicates whether the first frame
Rectified           = 0; %  Enable this flag to perform restricted motion sea

LFMotionSearchRange = 0; %  Enable this flag to adapt search range with respe
Horizontal_StepSize = 4; %  This flag specifies the horizontal distance betwe
Vertical_StepSize   = 4; %  This flag specifies the vertical distance between
```

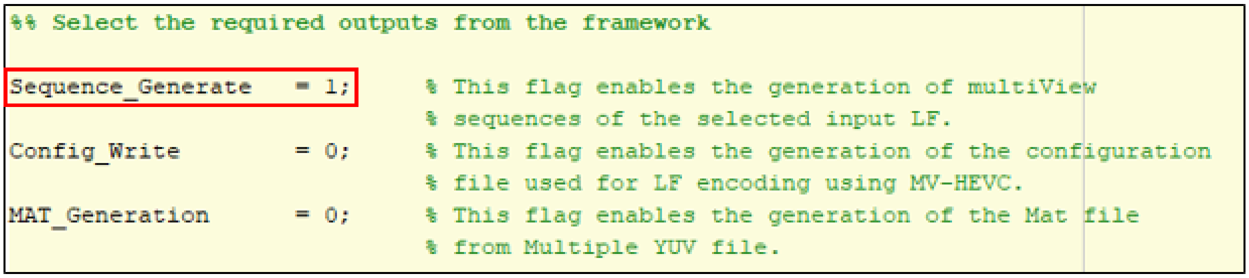
Fig. 4: Flags for control proposed motion optimization.

Description of these flags is as follows:

- **DummyFrame:** This flag indicates whether the first frame in the LF is a dummy frame. This flag must be enabled since we have used dummy frame in the proposed work to handle encoding of the central frame first. We are appending one extra frame in the beginning of LF views, since HEVC does not allow to encode any other frame but the first to be encoded at the start. If we manually reorder the frames of the input MPVS, then the AMVP module produces incorrect results. By including the dummy frame first, we can encode the central frame next to achieve the desired encoding order. The dummy frame in each view is Intra encoded, and at the end, its PSNR and bits are not utilized in the comparison.
- **Rectified**: Enable this flag to perform restricted motion search in horizontal or vertical directions (1D search), otherwise use default motion search (2D search)
- **LFMotionSearchRange**: Enable this flag to adapt search range with respect to maximum motion found in central LF column, otherwise use default search range (64 pixels)

- **Horizontal_StepSize**: This flag specifies the horizontal distance between adjacent cameras/lenses (required if LFMotionSearchRange == 1)
- **Vertical_StepSize**: This flag specifies the vertical distance between adjacent cameras/lenses (required if LFMotionSearchRange == 1)

Step 4. Generate multiple pseudo video sequence (MPVS) of the selected input LF by setting the *Sequence_Generate* flag to '1' as shown in Fig. 5, and execute the code. The output MPVS will be generated in the "/MPVS_Sequences/" folder.



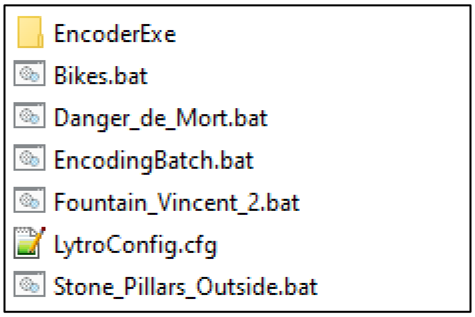
```
%% Select the required outputs from the framework

Sequence_Generate   = 1;     % This flag enables the generation of multiView
                             % sequences of the selected input LF.
Config_Write        = 0;     % This flag enables the generation of the configuration
                             % file used for LF encoding using MV-HEVC.
MAT_Generation      = 0;     % This flag enables the generation of the Mat file
                             % from Multiple YUV file.
```

Fig. 5: Parameters for specifying the required output to be generated using the framework.

Step 5. Generate the configuration file for encoding of the generated MPVS by setting *Config_Write* flag to '1' shown in Fig. 5, and execute the code. The output configuration file will be generated in the "/cfg/" folder. Place the generated configuration file in encoding system at path "/ProposedEncodingSystem/[Source]/".

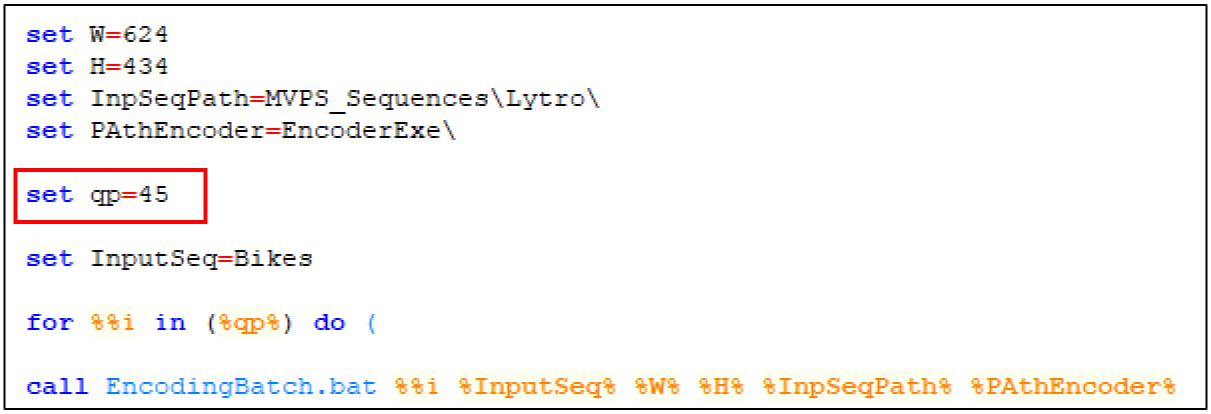
Step 6. Encode the generated MPVS using the proposed encoding system found in the "/ProposedEncodingSystem/[Source]/" folder for each source LF. The proposed encoding system contains executable batch files for each source LF, the MV-HEVC encoder executable, and the configuration file as shown in Fig. 6.



EncoderExe
Bikes.bat
Danger_de_Mort.bat
EncodingBatch.bat
Fountain_Vincent_2.bat
LytroConfig.cfg
Stone_Pillars_Outside.bat

Fig. 6: Contents of "/ProposedEncodingSystem/Lytro/" folder.

Step 6.1 The batch files ("**[LF].bat**") can be modified to specify the desired base QP (or a list of comma separated QPs) to encode the MPVS using the '*qp*' parameter shown in Fig. 7 (sample of "Lytro/Bikes.bat" LF batch file):



```
set W=624
set H=434
set InpSeqPath=MVPS_Sequences\Lytro\
set PAthEncoder=EncoderExe\

set qp=45

set InputSeq=Bikes

for %%i in (%qp%) do (

call EncodingBatch.bat %%i %InputSeq% %W% %H% %InpSeqPath% %PAthEncoder%
```

Fig. 7: Batch file of Lytro/Bikes LF.

Step 6.2 To start the encoding process, run the batch file corresponding to the source LF. The encoded bitstream and reconstructed YUV files will be generated in the "/ProposedEncodingSystem/[Source]/Output_[source]/" folder.

Step 7. Compressed MAT files can be generated from the reconstructed YUV files of the MPVS using the third flag, *Mat_Generation* shown in Fig. 5 in step 4 above. The MAT files will be generated in the "/ Compressed_Mats/" folder. During MAT file generation the script also perform PSNR and SSIM estimation as defined in JPEG pleno common test conditions. The Mat files contains LF structure, PSNR_Y_mean, PSNR_YUV_mean, SSIM_mean and Rate in bytes for compressed LF.

The **Write_PPM** flag will enable the writing of decoded views in .PPM format in similar to Groundtruth format. (Lytro 10 bpp, Franhoufer 10 bpp and Stanford 8 bpp)

Summary

The proposed coding framework reads the input LF images, generates Multiview Sequences, encodes the generated sequences and finally produces decoded Mat files as LF structure along bitrate in Bytes.

==

If there are any issues with the usage of the LF coding framework, please contact Waqas Ahmad <Waqas.Ahmad@miun.se>.