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

ISO/IEC JTC1/SC29/WG11 MPEG2018/m46126
January 2019, Marrakesh, Marocco

Title	**Depth Estimation Reference Software extension for lightfield images**
Status	**Input**
Source	**Poznań University of Technology**
Authors	**Krzysztof Wegner (krzysztof.wegner@put.poznan.pl),**
	Tomasz Grajek,
	Adam Grzelka,
	Mateusz Lorkiewicz,
	Robert Ratajczak
	Olgierd Stankiewicz,
	Jakub Stankowski
	Hubert Żabiński,
	Marek Domański

1. Introduction

Recently extension of Depth Estimation Reference software was proposed [8], which searches for correspondence towards four reference views, which allows depth estimation based on 5 views in total. Although such configuration may be satisfactory in the case of multiview video, it is not sufficient for even the simplest lightfield images, which can be composed from tens of views. Estimation of a depth map for such a content using only 5 views does not exploit all of the information available in all of the views. This document reports extension of Depth Estimation Reference Software, which allows depth map estimate based on multiple (arbitrary number) input views. Such an approach allows substantial increase of the quality of estimated depth, especially for the scenes with significant amount of occlusions. For a simple case of three input views, the proposed version of Depth Estimated Reference Software works exactly the same as version 6.1 [7] on which it is based upon.

Moreover, the source code of Depth Estimation Reference software has been refactored, and cleaned up to allow future modifications. Other minor improvements include:

- Generalization of Block Matching procedure - to allow other similarity matrices to be implemented,
- Generalization of Optimization procedure to allow not only Graph Cut optimization to be implemented,
- Bug fixes allowing compilation under Linux,
- Bug fixes for DepthType parameter.

2. Developed extensions and modifications to DERS v6.1

General overview of the depth estimation algorithm implemented in Depth Estimation Reference Software version 6.1 is presented in Fig 1. Three input views (left, center and right) are fed into Block Matching block. Based on the candidate depth values and pre-calculated locations of corresponding fragments in the left, center and right input images, Block Matching block calculates matching errors between the left and the center view and the right and center view. As a result, Left and Right Matching Error Volume is created. The comparison method of the corresponding blocks in the input images, and the size of the blocks can be selected in configuration file.

Additionally Center View Image is segmented in Segmentation block. The particular segmentation method can be selected from several methods in the configuration file. As a result image containing indices of a segments containing each pixel of center view is created. This information may be used during Graph Cuts optimization (which is performed on pixels) to control smoothing coefficient so that depth within segments is smooth and may change rapidly in between of segments.

Two Matching Error Volumes, center view and Segment Index Map for the center view are fed into a Graph Cut Optimization routine, which performs actual depth selection for each pixel of a the center view.

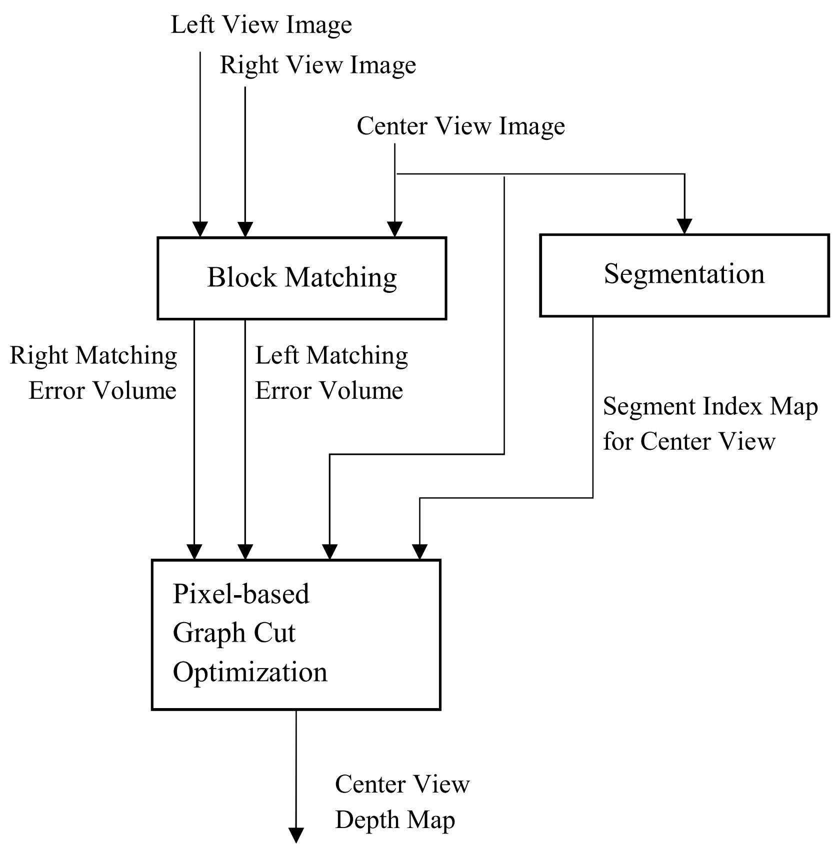

Fig. 1. Block diagram of the algorithm implemented in
Depth Estimation Reference Software version 6.1 (base software version of the proposal).

In the proposed version of Depth Estimation Reference Software, the implementation of the algorithm has been refactored in order to allow simultaneous estimation of multiple depth maps from multiple input views images. The block diagram of the generalized algorithm has been presented in Fig. 2. At the input of the processing, multiple images of the input views are provided. Some of the input views (e.g. all) can be selected as output views for which depth maps will be estimated. The selected set of the input views images and the selected output views are feed into a Block Matching module. Block Matching also has been generalized in the proposal: based on N input views and M output views N·M matching error volumes are generated (instead of two: for the left and for the right view). All previously implemented routines for block matching have been simplified to handle only one input and one output view (previously the right or the left and the center view) and reused in the newly implemented meta-block matching routine (Fig. 3). Also the set of segmentation data has been generalized in order to handle multiple output views. Similarly like in block matching, new segmentation routine reuse the Old Block Matching routines from DERS 6.1, and simply loops over all of the output views instead of only one output view as before.

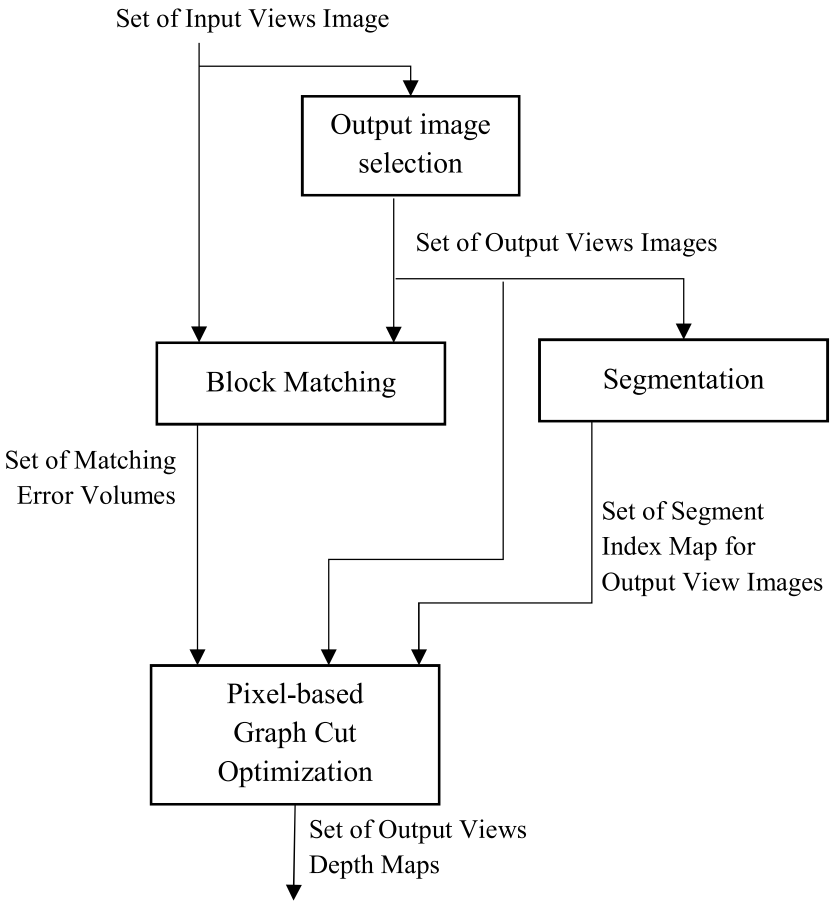

Fig. 2. Block diagram of the proposed algorithm.

Moreover, the source code of the graph cut optimization algorithm has been refactored and new meta-optimization procedure has been created. It allows for flexible specification of optimization function. For now, it uses old routines and old optimization function, but it is prepared for the use of new more advanced ones that in future may exploit e.g. the fact of simultaneous multiple depth estimation.

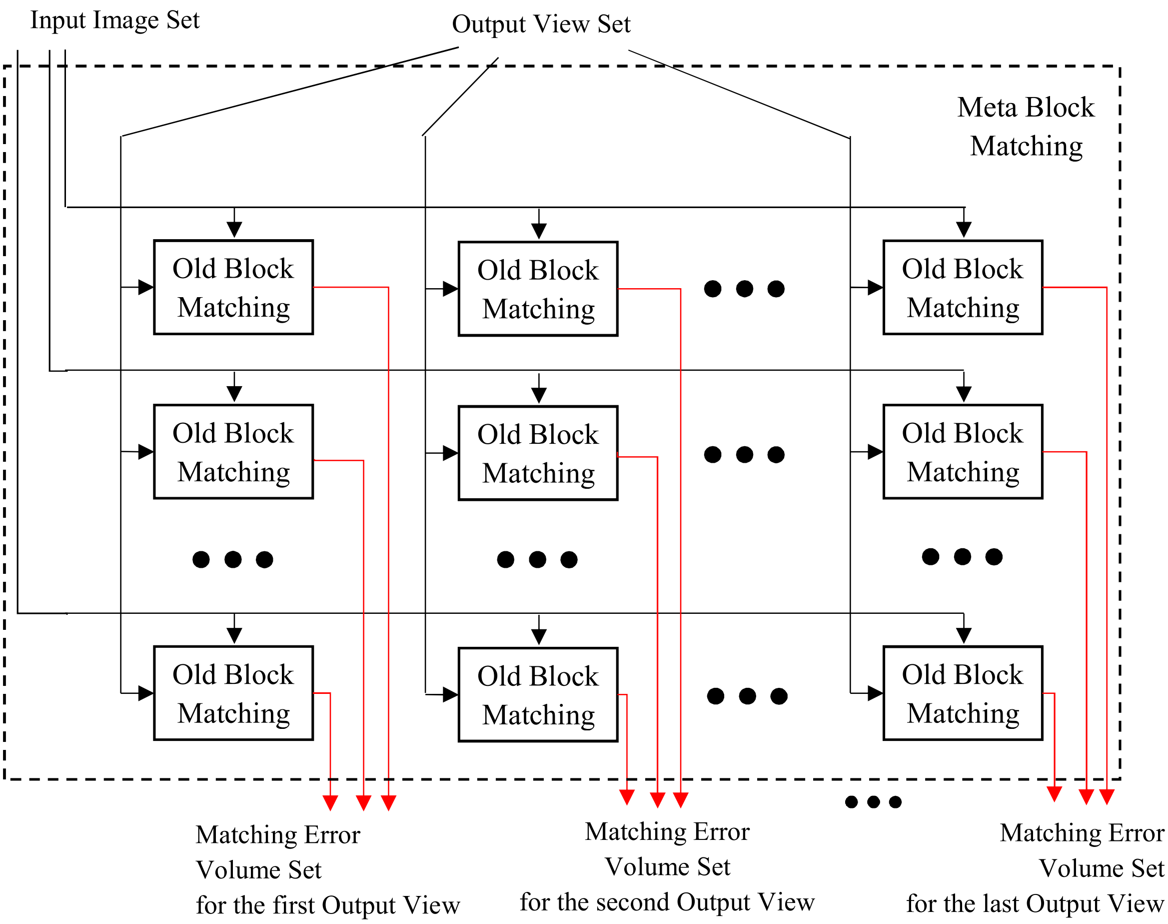

Fig. 3. New meta block matching procedure composed of Old Block Matching routine (from unaltered DERS 6.1).

3. View naming convention

The proposed extension allows for theoretically unlimited number of input views (practically limited by the available memory volume) and the corresponding depth maps. Instead of using names: Left, Center and Right for consecutive: camera names, input views, depth maps (and other parameters related to given view), all of these are now numbered with indices.

In particular, instead of old camera parameter names, which are based on left/center/right naming convention, e.g.:

LeftCameraName
CenterCameraName
RightCameraName

new naming convention has been introduced, which is based on indexes of cameras, views etc. For example

InputCameraName0
InputCameraName1
InputCameraName2

Example of the configuration file:
New configuration parameters have been highlighted in red. Configuration parameters that have been changed are marked in blue.

```
#================ GENERAL PARAMETERS ================
SourceWidth                   1920
SourceHeight            1080

StartFrame             160
TotalNumberOfFrames    1

FileCameraParameter    Calib_Param_Kermit.txt

DepthType              0          # 0...from camera, 1...from world origin

#================ KAMERA CONFIGURATION ================

NumberOfInputViews     5       # Number of input views based on which depths will be estimated
NumberOfOutputDepths   1       # Number of output views for which depth will be estimated

InputCameraName0       param_v6  # Camera parameter name of input view 0
InputCameraName1       param_v7
InputCameraName2       param_v8
InputCameraName3       param_v5
InputCameraName4       param_v9

OutputCameraName0      param_v7  # Camera parameter name of output view 0

#================ INPUT PARAMETERS ================

FileInputViewImage0     ..\TextureContent\v6_1920x1080_8bps_YUV420.yuv
FileInputViewImage1     ..\TextureContent\v7_1920x1080_8bps_YUV420.yuv
FileInputViewImage2     ..\TextureContent\v8_1920x1080_8bps_YUV420.yuv
FileInputViewImage3     ..\TextureContent\v5_1920x1080_8bps_YUV420.yuv
FileInputViewImage4     ..\TextureContent\v9_1920x1080_8bps_YUV420.yuv

#================ OUTPUT PARAMETERS ================

FileOutputDepthMapImage0 v7_1920x1080_0_3_1_62d_16bps_cf400.yuv

#================ OLD UNUSED PARAMETERS ================

LeftCameraName         param_v6
CenterCameraName       param_v7
RightCameraName        param_v8
FileLeftViewImage       ..\TextureContent\v6_1920x1080_8bps_YUV420.yuv
FileCenterViewImage     ..\TextureContent\v7_1920x1080_8bps_YUV420.yuv
FileRightViewImage      ..\TextureContent\v8_1920x1080_8bps_YUV420.yuv
FileOutputDepthMapImage       v7_1920x1080_0_3_1_62d_16bps_cf400.yuv

#================ SEARCH PARAMETERS ================

SearchRangeType                       0            # 0...Min/MaxDisparity, 1...Znear/Zfar
MinimumValueOfDisparitySearchRange    35
MaximumValueOfDisparitySearchRange    190
MinimumValueOfDisparityRange          35
MaximumValueOfDisparityRange          190

NearestDepthValue                     0.29
FarthestDepthValue                    1.63
NearestSearchDepthValue               0.29
FarthestSearchDepthValue              1.63
NumberOfDepthSteps                    50


SmoothingCoefficient                  3.8          # SC1 for non-edge
SmoothingCoefficient2                 0.8          # SC2, SC1*SC2 for edge

BaselineBasis                         0     # cam distance from 0...min of left or right,
                                            #                   1...max of left or right,
                                            #                   2...left, 3...right
```

```
Precision                  1  #horizontal precision, 1...integer, 2...half, 4...quarter
VerticalPrecision          1  #vertical   precision, 1...integer, 2...half, 4...quarter

Filter                     0  # upsample horizontal filter: 0...bi-linear, 1...bi-cubic
                              # and 2...MPEG-4 AVC 6-tap.
VerticalFilter             0  # upsample vertical   filter: 0...bi-linear, 1...bi-cubic
                              # and 2...MPEG-4 AVC 6-tap support upto 1/4pixel.

SearchLevel             1     # 1...integer, 2...half, 4...quarter, 8...1/8 pixel

MatchingMethod          0     # Homography: 0...conventional, 1...disparity-base,
                              #             3...soft-segmentation-base

#========== Size of Matching Block (ignored when MatchingMethod = 3) ==========
MatchingBlock           1     # 1...pixel matching, 3...3x3 block matching

#========== Segmentation ==========
ImageSegmentation       0   # 0...off, 1...on difficult to use
SegmentationMethod      1   # 1...mean shift, 2...phyramidal, 3...kernel clustering
MaxCluster              10  # positive integer value

#========== Temporal Enhancement ==========
TemporalEnhancement     0     # acceleration: 0...off, 1...on skips no-motion pixel in frame
Threshold               1.00  # motion detection, pixel difference in two frames

#========== Semi-automatic Depth Estimation ==========
DepthEstimationMode     0     # 0...automatic (Graph-Cuts). 1...semi-automatic 1,
                              # 2...semi-automatic 2, 3...reference depth mode.

ReliabilityThreshold    0     # rTh. if texture slope S < rTh, matching error is
                              # weighted by *rTh/S
SmoothingThreshold      12    # sTh. if texture slope S < sTh, depth smoothing
                              # coefficient SC1 is weighted by *SC2*sTh/S

#---- For DepthEstimationMode = 1 or 2  ------
FileCenterManual                #Path and filename prefix of the manual input files

#---- For DepthEstimationMode = 2  ------
ThresholdOfDepthDifference      # Threshold value of depth difference
MovingObjectsBSize              # 0: small  1: medium  2: large
MotionSearchBSize               # 0: narrow 1: medium 2: wide

#---- For DepthEstimationMode = 3  ------
RefDepthCameraName              # camera parameter name
RefDepthFile                    # filename reference depthmap video
```

4. Memory footprint

In DERS 6.1 [7], correspondence search was performed in the right and in the left view with respect to the provided center view. This was resulting in 2 matching error volumes (marching errors in function of pixel position and disparity: x,y,d) – one for the left view and one for the right view. The implemented, proposed extension employs matching of N input views with M output views simultaneously. Therefore, in in general it requires $N \cdot M / 2$ times more memory. For example, estimation of one depth map from 16 views current implementation requires more than 20 GB of RAM. It has to be noticed that 16 views lighfield is rather small one. In order to allow at least moderate lightfield images composed from hundreds of views to be processed, memory consumption of the algorithm implemented in DERS needs to be vastly reduced in future releases.

5. Implementation details

Multi-input-view support has been implemented on top of DERS 6.1. All new introduced source code has been encapsulated in conditional code block, enabled by `MULTI_INPUTVIEWS` and `MULTI_OUTPUTDEPTHS` compilation flags.This allows for easy switching on and off of the proposed extension, during compilation.

Due to memory savings, there is a compilation flag (`MULTI_INPUTVIEWS_LENGTH`), which allows to set the maximum number of input views allowed. Similarly, the compilation flag (`MULTI_OUTPUTDEPTHS_LENGTH`) allows to set the maximum number of output depth maps to be estimated.

6. Results

The proposed version of DERS has been compared against “Enhanced DERS for Quad Reference Views” [8]. Both software versions have been compiled in Visual Studio 2017 with OpenCV 3.4.0 under 64-bit Microsoft Windows.

Experiments have been done according to "Exploration Experiments for MPEG-I: Windowed-6DoF [11] for “Technicolor Painter” data set. The latest configuration files for DERS have been downloaded from MPEG content repository. Configuration files for the proposed enhanced DERS have been created based on configuration files for DERS 6.1 [7], and are attached to this contribution. In the case of the proposed extension, the depth maps have been estimated from all 16 views of the “Technicolor Painter”.

Quality of the depth maps has been evaluated indirectly by means of quality of the views synthesized with VSRS 5.0. The quality of the synthesized view is provided in Table 1. For some of the virtual views, slight improvement has been observed caused by usage of depth maps estimated with the proposed extension.

Table 1. Quality of the virtual views rendered from two input views obtain for "Technicolor Painter" dataset. Configuration of the renderer according to [11].

Synthesized View	Input Views		Anchor Depth Maps	New Depth Maps
			PSNR [dB]	PSNR [dB]
10	5	15	29.38	29.42
10	6	14	29.53	29.51
10	9	11	30.18	30.20
11	7	15	33.15	33.17
13	12	14	31.40	31.37
14	13	15	31.09	31.09
2	1	3	35.09	35.07
4	0	8	35.04	34.99
5	0	10	33.65	33.62
5	1	9	35.99	35.94
5	4	6	34.36	34.41
6	1	11	32.97	33.00
6	5	7	32.54	32.58
7	4	12	31.13	31.11
8	4	12	34.27	34.22
9	4	14	32.78	32.83
9	5	13	35.19	35.17
9	8	10	31.70	31.72

7. Conclusions

A new extension for multi reference depth estimation based on DERS 6.1 has been proposed. Its purpose is to support depth estimation in cases applicable for lightfields, therefore for content consisting of many views, e.g. more than 16.

The proposal allows for depth estimation from unlimited number of input views. Moreover it can be used to estimate depth for more than one output view simultaneously. Further study of the performance of new DERS version is needed. Moreover memory footprint reduction is desired to allow bigger lightfiled images to be processed with the software.

The authors believe that it would be advantageous to merge this proposed extension with the versions of DERS (including current version of eDERS [9]) currently under consideration by MPEG-I.

8. Acknowledgment

Project funded by the National Centre for Research and Development in the LIDER Programme (LIDER/34/0177/L-8/16/NCBR/2017)

9. References

[1] M. Tanimoto, T. Fujii, and K. Suzuki, "Multi-view depth map of Akko & Kayo and Rena", ISO/IEC JTC1/SC29/WG11, M14888, October 2007.

[2] M. Tanimoto, T. Fujii and K. Suzuki, "Improvement of Depth Map Estimation and View Synthesis", ISO/IEC JTC1/SC29/WG11, M15090, January 2008.

[3] M. Tanimoto, T. Fujii and K. Suzuki, "Reference software for Depth Estimation and View Synthesis", ISO/IEC JTC1/SC29/WG11, M15377, April 2008.

[4] M. Tanimoto, T. Fujii and N. Fukushima, "Depth Estimation using Stereo Camera Rectification for Non-parallel Camera Setup", ISO/IEC JTC1/SC29/WG11, M15380, April 2008.

[5] S. Shimizu and H. Kimata, "Experimental Results on Depth Estimation and View Synthesis with sub-pixel precision," ISO/IEC JTC1/SC29/WG11, M15584, July 2008.

[6] M. Tanimoto, T. Fujii and K. Suzuki, "Reference Software of Depth Estimation and View Synthesis for FTV/3DV," ISO/IEC JTC1/SC29/WG11, M15836, October 2008.

[7] Krzysztof Wegner, Olgierd Stankiewicz, Masayuki Tanimoto, and Marek Domanski, "Enhanced Depth Estimation Reference Software (DERS) for Free-viewpoint Television", ISO/IEC JTC1/SC29/WG11, M31518, October 2013, Geneva, Switzerland.

[8] Takanori Senoh, Kenji Yamamoto, Nobuji Tetsutani, and Hiroshi Yasuda, "Enhanced DERS for Quad Reference Views (eDERS)", ISO/IEC JTC1/SC29/WG11, M41955, January 2018, Gwangju, Korea.

[9] Takanori Senoh, Nobuji Tetsutani, and Hiroshi Yasuda, "Updated eDERS to Higher Precision", ISO/IEC JTC1/SC29/WG11, M42525, April 2018, San Diego, USA.

[10] Takanori Senoh, Nobuji Tetsutani, Hiroshi Yasuda, and Mehrdad Teratani, "Proposed Depth Estimation Reference Software (pDERS8) Manual", ISO/IEC JTC1/SC29/WG11, M44030, October 2018, Macau, China.

[11] Eduardo Juárez, Didier Doyen, Gauthier Lafruit , "Exploration Experiments for MPEG-I: Windowed-6DoF", ISO/IEC JTC1/SC29/WG11, N18065, October 2018, Macau, China.