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

ISO/IEC JTC1/SC29/WG11 MPEG2014/M34302
July 2014, Sapporo, Japan

Source	**Poznan University of Technology,**
	Chair of Multimedia Telecommunications and Microelectronics
Status	**Contribution**
Title	**DERS Software Manual**
Author	Krzysztof Wegner (kwegner@multimedia.edu.pl)
	Olgierd Stankiewicz

Introduction

Version:	DERS 6.1 (SVN tag: ders-6.1)
Last update:	January 16, 2014

Summary:

This document contains a detailed description of the usage and configuration of DERS (*D*epth *E*stimation *R*eference *S*oftware) for the Free viewpoint TeleVision (FTV) project of the ISO/IEC Moving Pictures Experts Group (MPEG).

Software is a modified and extended version of the software used earler during second phase of FTV development

This paper provides information how to build the software on Windows and Linux platforms. It contains a description of the usage and configuration for the binaries built from the software package, including examples for depth estimation scenarios.

Table of Contents

1 General Information

The DERS is the reference software for the FTV project of the ISO/IEC Moving Pictures Experts Group (MPEG). Since the FTV project is still under development, the DERS is also under development and changes frequently.

The DERS is written in C++ and is provided as source code. Section 1.1 describes how the DERS can be obtained via a SVN server. Information about the structure of the MPEG SVN repository is presented in section 1.2. Section 1.3 describes how the DERS can be built on Windows and Linux platforms.

1.1 Accessing the latest DERS

In order to keep track of the changes in software development and to always provide an up-to-date version of the DERS, a SVN server for the DERS has been set up at the MPEG sc29wg11 website. The SVN server can be accessed using Tortoise SVN or any other SVN client. The server is configured to allow read access only using the parameters specified in Table 1. Write access to the DERS server is restricted to the DERS coordinators group.

Table 1: SVN access parameters

trunk url address:	http://wg11.sc29.org/svn/repos/Explorations/FTV/DERS/trunk/
tag url address:	http://wg11.sc29.org/svn/repos/ Explorations/FTV/DERS/tags/DERS-6.1
user name:	sc29wg11
password:	same password as MEPG account, may be updated every few months

Example 1 shows how the DERS can be accessed by using a command line SVN client.

Example 1: Accessing the DERS with a command line SVN client

```
svn co http://wg11.sc29.org/svn/repos/Explorations/FTV/DERS/trunk DERS
```

In Example 2, it is shown how a specific DERS software version – specified by a tag (DERS-5.1 in Example 2) – can be obtained using a command line SVN client. Note that *co* represents an abbreviation for the command *checkout*, which was used in Example 1.

Example 2: Accessing the DERS software version with the tag DERS-5.1 with a command line SVN client

```
svn co http://wg11.sc29.org/svn/repos/Explorations/FTV/DERS/tags/DERS-5.1 DERS-5.1
```

1.2 Structure of the MPEG SVN Repository

After accessing the DERS as described in section 1.1, a folder *DERS* is created. The directory structure of this folder is summarized in Table 2. The folder *DERS* contains all files that are required for building and running the software except OpenCV library which must be downloaded separately (see section 1.5).

Source code of the DERS is divided into three parts: CommonLibStatic, DepthEstLibStatic and DepthEst.

The folder *DERS/CommonLibStatic* is structured into sub-folders for all source (*.cpp) and include (*.h) files for the common libraries of the VSRS and DERS. The folder *DERS/DepthEstLibStatic* is structured into sub-folders for all source and include files for the libraries of the DERS. The folder

DERS/DepthEst is structured into sub-folders for all source and include files for the application of the DERS.

A log file describing the main changes from one DERS software version to the next is given by *ders_changes.txt*. Note that this log files starts with the Depth Estimation version 0.

Table 2: Structure of the MPEG SVN repository for the DERS

Folder	***Content***
DERS	*source code and project files for the DERS* *All files that are required for building and using the VSRS software are contained in this folder.*
DERS/windows	*workspaces* *Workspaces are provided for Microsoft Visual Studio 6 and Microsoft Visual Studio .NET.*
DERS/linux	*makefiles* *Makefiles are provided for Linux*
DERS/DepthEst	*source code and project files for the DERS applicaton* *All files that are required for building the DERS application are contained in this folder.*
DERS/DepthEstLibStatic	*source code and project files for the DERS library* *All files that are required for building the DERS libraries are contained in this folder.*
DERS/CommonLibStatic	*source code and project files for the VSRS and DERS common library* *All files that are required for building the common libraries of the VSRS and DERS are contained in this folder.*
DERS/camera_parameter_files	*camera parameter files*
DERS/configuration_files	*configuration files*
DERS/bin	*location of binaries after building the software* *More information about the binaries are given in section 1.4*
DERS/lib	*location of libraries after building the software* *More information regarding the libraries are given in section 1.4*
DERS/doc	*changes log file and software manual* *ders_changes.txt file described the (main) changes from one SVN version to the next. It starts with DepthEstimations version 0 (SVN tag: DE0).*

1.3 Building the DERS

It shall be possible to build the DERS on a Windows platform with Microsoft Visual Studio version from 6 to 11 and on a Linux platform with gcc. For information on how to build the software on a Windows platform with Microsoft Visual Studio, refer to section 1.3.1, and for information on how to build the software on a Linux platform with gcc refer to section 1.3.2.

Since the DERS is written in C++, it should also be possible to build the software on other platforms, which provide a C++ compiler. However, it is only guaranteed that the software can be build by using Microsoft Visual Studio version 6 to 11 or the gcc compiler.

1.3.1 Windows platform with Microsoft Visual Studio

The folder *DERS/windows* contains a Microsoft Visual Studio 2012 solution *DERSVC11.sln*. In order to build the software, open this workspace with Microsoft Visual Studio 2012, and build the project file by selecting *Build → Build Solution*.

After building the software the folders *bin* shall contain the executable binaries *DERS/bin/DepthEst.exe* and *DepthEstd.exe*. The folders *CommonLibStatic/Debug* and *CommonLibStatic/Release* contain libraries *DERS/CommonLibStatic/Debug/CommonLibStatic.lib* and *DERS/CommonLibStatic/Release/CommonLibStatic.lib*, respectively. The folders *DepthEstLibStatic/Debug* and *DepthEstLibStatic/Release* contain libraries *DERS/DepthEstLibStatic/Debug/DepthEstLibStatic.lib* and *DERS/DepthEstLibStatic/Release/DepthEstLibStatic.lib*, respectively. Note that there exist two different versions for the binary, one with and one without a "d" before the dot. The version with a "d" before the dot represent binary that has been built in debug mode, while the version without a "d" before the dot represent binary that has been built in release mode.

The folder *DERS/windows* also contains a Microsoft Visual Studio 2010 solution *DERSVC10.sln*. In order to build the software, open this workspace with Microsoft Visual Studio 2010, and build the project file by selecting *Build → Build Solution*.

The folder *DERS/windows* also contains a Microsoft Visual Studio .NET 2003 workspace *DERSVC7.sln*. However, the compilation under Microsoft Visual Studio .NET 2003 is only occasionally checked, and thus it is not guaranteed that each software version can be build using Microsoft Visual Studio .NET 2003. In order to build the software, open this workspace with Microsoft Visual Studio .NET 2003, and build the project file by selecting *Build → Batch build*.

The folder *DERS/windows* also contains a Microsoft Visual Studio 6 workspace *DERSVC6.dsw*. However, the compilation under Microsoft Visual Studio 6 is only occasionally checked, and thus it is not guaranteed that each software version can be build using Microsoft Visual Studio 6. In order to build the software with Microsoft Visual Studio 6, open the workspace with Microsoft Visual Studio 6.0, and build the project file by selecting *Build→ Batch build*, which opens a new dialog window. Then press the buttion *Select All* and *Rebuild*.

1.3.2 Linux platform with gcc compiler

Makefiles for the Linux with gcc compiler is provided in the folder *DERS/linux* and corresponding sub-folders. For example, if the currect folder is the folder *DERS/lunux* of the DERS repository (see section 1.2), the command specifies in Example 3 should be executed to build all project files.

Example 3: Building the DERS under Linux with a gcc compiler.

```
Make
```

By replacing *make* with *make release* or *make debug* in the Example 3, it can be specified that only the release or debug versions of the libraries and executables should be built.

After building the software the folders *bin* and *lib* shall contain the binaries and libraries *DERS/bin/DepthEst, DepthEstd* and libraries *DERS/lib/libDepthEstLibStatic.a, libDepthEstLibStaticd.a*, *DERS/lib/libCommonLibStatic.a, libCommonLibStaticd.a*, respectively. Note that there exist two different versions for each binary or library, one with and one without a "d" before the dot. The versions with a "d" before the dot represent binaries or libraries that have been built in debug mode, while the versions without a "d" before the dot represent binaries or libraries that have been built in release mode. When the command *make release* or *make debug* was used, only the debug or release version are present, respectively.

1.4 Information on binary and library

Table 3 and Table 4 give information on the library and executable that are contained in the DERS software package. Note that – as described in sections 1.3.1 and 1.3.2 – the naming of the actual library and executable files is dependent on the platform and on whether the library and executable have been built in release or debug mode.

Detailed information on command line options and configuration parameters for the executables are given in section 2. And in section 2 and examples for using the DERS software are given in the form of a tutorial. And in section 0 an example perl script is given to run batch simulation tasks with DERS.

Table 3: Library provided by the VSRS software

library	***Description***
DepthEstLibStatic	*Depth estimation algorithm lib* This library provides classes that are used to do depth estimation.
CommonLibStatic	*common lib* This library provides classes that are used by both VSRS and DERS softwares.

Table 4: Executable provided by the VSRS software

executable	***Description***
DepthEst	*Depth estimation application* The application to run depth estimation tasks. More information on the usage of the program is provided in section 2.

1.5 OpenCV

To compile DERS, the OpenCV library must be installed on your computer. The version of the OpenCV has to be 2.4.6 or more. If you are not familiar with OpenCV, refer

http://opencv.org/

If opencv-devel is installed using yum of Fedora, opencv ver.2.4.6 or more is installed automatically. If opencv ver.2.4.6 or more is not installed, please download the source files from

http://sourceforge.net/projects/opencvlibrary/

and compile and install them.

The header files of opencv have to be set by user. If pkg-config is installed and opencv-devel is configured in user’s PC, makefiles of DERS do not need any changes.

2 Usage and configuration of the DERS

This section provides information on usage and configuration of the binary contained in the DERS package. Additionally, examples for using the software are provided in section 0.

2.1 DERS

The DERS can be used for generating intermediate views. The basic Depth Estimation call is illustrated in Example 4 and Example 5. At this *cfg* represents the filename of the configuration file. The configuration file shall be specified for each DERS call.

Example 4: Using the DERS with Visual Studio 2012

```
DERSVC11.exe <cfg>
```

Example 5: Using the DERS with Visual Studo 2010

```
DERSVC10.exe <cfg>
```

Generally, the configuration files present a collection of configuration parameters. Each configuration parameter is specified in one line of the configuration file. Comments are started by the character '#'. The order of configuration parameters inside a configuration file can be arbitrarily selected. Each configuration parameter has a default value, and when the configuration parameter is not present in the configuration file, the default value is taken instead.

Thus, it is generally not required to specify all configuration parameters in a configuration file. Finally, it should be noted that the configuration parameter values specified in the configration file can be replaced by the values specified via command line options.

Two example configuration files are provided in the following sub-sections.

2.2 Configuration file

All available configuration file parameters for the depth estimation together with a brief description are summarized in Example 6. Additional information about the configuration parameters is given below.

Example 6: DERS configuration file

```
#=============== INPUT PARAMETERS ===============
DepthType                             0               # 0...Depth from camera,
1...Depth from the origin of 3D space
SourceWidth                           1280            # Input frame width
SourceHeight                          960             # Input frame height
StartFrame                            0               # Starting frame #
TotalNumberOfFrames                   300             # Total number of input
frames
LeftCameraName                        param_dog37     # Name of left camera
CenterCameraName                      param_dog38     # Name of center camera
RightCameraName                       param_dog39     # Name of right camera
MatchDirection                        3                # 1...Left 2...Right
3...Both
MinimumValueOfDisparitySearchRange    0               # Minimum value of disparity
search range. This value is not always same as all pairs of views.
MaximumValueOfDisparitySearchRange    20              # Maximum value of disparity
search range. This value is not always same as all pairs of views.
MinimumValueOfDisparityRange          0               # Minimum value of disparity
range. This value is smaller than or equal to minimum value of disparity search
ranges for all views.
MaximumValueOfDisparityRange          20              # Maximum value of disparity
range. This value is larger than or equal to maximum value of disparity search
ranges for all views.
SmoothingCoefficient                  2.0             # Smoothing coefficient to
compute depth maps
FileLeftViewImage        C:\\YUV\\Dog\\dog037.yuv      # Name of left input video
FileCenterViewImage      C:\\YUV\\Dog\\dog038.yuv      # Name of center input video
FileRightViewImage       C:\\YUV\\Dog\\dog039.yuv      # Name of right input video
FileOutputDepthMapImage  depth_dog038.yuv             # Name of output depth map
file
FileCameraParameter    ..\\camera_parameter_files\\cam_param_dog.txt    # Name of
text file which includes camera parameters

BaselineBasis            1  # 0...minimum baseline, 1...maximum baseline,
                              2...left baseline, 3...right baseline
Precision                4  # 1...Integer-pel, 2...Half-pel, 4...Quater-pel
VerticalPrecision        4  # 1...Integer-pel, 2...Half-pel, 4...Quater-pel
SearchLevel              2  # 1...Integer-pel, 2...Half-pel, 4...Quater-pel
Filter                   1  # 0...(Bi)-linear, 1...(Bi)-Cubic, 2...MPEG-4 AVC 6-tap
VerticalFilter           1  # 0...(Bi)-linear, 1...(Bi)-Cubic, 2...MPEG-4 AVC 6-tap
MatchingMethod           2  # 0...Conventional, 1...Disaprity-based,
                              2...Homography-based, 3...Soft-segmentation-based
                              4...Epipolar-line-based

#========== Temporal Enhancement ==========
TemporalEnhancement   1     # 0...Off, 1...On : Temporal Enhancement of Depth Est.
```

```
Threshold              2.00  # Threshold of MAD

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

#========== Softsegmentation (for MatchingMethod = 3 only) ==========
SoftDistanceCoeff    10.0    # SoftSegmentation Distance Coefficient
SoftColorCoeff       20.0    # SoftSegmentation Color Coefficient
SoftBlockWidth       11      # SoftSegmentation Block Width
SoftBlockHeight      11      # SoftSegmentation Block Height

#========== Segmentation ==========
ImageSegmentation       0    # 0...Off, 1...On
SmoothingCoefficient2  1.00 # Smoothing coefficient to compute depth maps
SegmentationMethod      3    # 1...mean shift algorithm, 2...phyramid segmentation,
3...K mean clustering
MaxCluster             32    # Positive Integer Value

#========== Occlusions handling =============
Occlusion               1    # 0...Off, 1...On

#========== Semi-automatic Depth Estimation ==========
DepthEstimationMode      1   #0...automatic Depth Estimation; 1...Semi-automatic
mode1; 2...Semi-automatic mode2; 3...Reference Depth mode

#---- For DepthEstimationMode = 1  ------
FileCenterManual        dog_38  #Path and filename prefix of the manual input files

#---- For DepthEstimationMode = 2  ------
ThresholdOfDepthDifference  10  # Threshold value of depth difference
MovingObjectsBSize           1  # 0: small  1: medium  2: large
MotionSearchBSize            0  # 0: narrow 1: medium 2: wide

#---- For DepthEstimationMode = 3  ------
RefDepthCameraName      param_dog39         # camera param
RefDepthFile            rdepth_dog_39.yuv   # filename reference depthmap video
```

DepthType

Bool (0 or 1), default: 0

Specifies the depth type. The input value 0 means the mode computing depth from a camera. The input value 1 means the mode computing depth from the origin of 3D space.

SourceWidth

Unsigned Int, default: 1280

Specifies the width of the input images. *SourceWidth* shall be a positive.

SourceHeight

Unsigned Int, default: 960

Specifies the height of the input images. *SourceHeight* shall be a positive.

StartFrame

Unsigned Int, default: 0

Specifies the start frame number. *StartFrame* shall be a nonnegative.

TotalNumberOfFrames

Unsigned Int, default: 300

Specifies the number of frames of the input sequence to be computed depth. TotalNumverOfFrames shall be a positive integer.

LeftCameraName

String, default: param_dog36

Specifies the name of left camera to be used for depth estimation.

`CenterCameraName`

String, default: param_ dog38

Specifies the name of center camera to be computed depth map.

`RightCameraName`

String, default: param_ dog40

Specifies the name of right camera to be used for depth estimation.

`MatchDirection`

Unsigned Int, (1 or 2 or 3), default: 3

Specifies the direction of correspondence search. 1 means use only left and center view for depth estimation, 2 means use only right and center view for depth estimation, 3 means use all view.

`MinimumValueOfDisparitySearchRange`

Int, default: 0

Specifies the minimum value of disparity search range. *MinimumValueOfDisparitySearchRange* shall be an integer.

`MaximumValueOfDisparitySearchRange`

Int, default: 20

Specifies the maximum value of disparity search range. *MaximumValueOfDisparitySearchRange* shall be an integer.

`MinimumValueOfDisparityRange`

Int, default: 0

Specifies the minimum value of disparity range. *MinimumValueOfDisparityRange* shall be an integer. This value corresponds to the farthest depth value.

`MaximumValueOfDisparityRange`

Int, default: 20

Specifies the maximum value of disparity range. *MaximumValueOfDisparityRange* shall be an integer. This value corresponds to the nearest depth value.

Disparity search range is not always same as all pairs of views. Changing this range may effects resultant depth maps. This range is determined visually. Disparity range has to include disparity search ranges for all views. Minimum value of disparity range is smaller than or equal to minimum value of all disparity search ranges. In the same way, maximum value of disparity range is larger than or equal to maximum value of all disparity search ranges. Minimum and maximum values of disparity range correspond to Z_{far} and Z_{near} values. Figure 1 shows this relation.

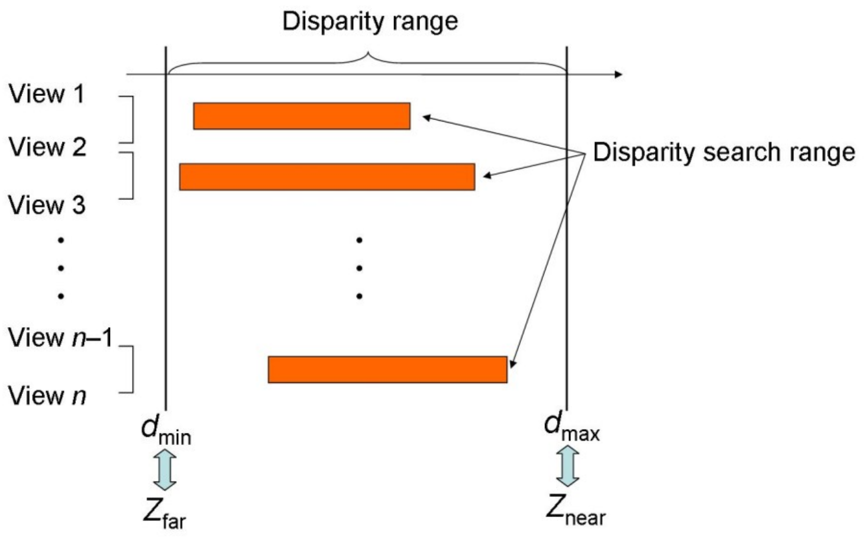

Figure 1: Relation between disparity range and disparity search range

`SmoothingCoefficient`

Double, default: 4.0

Specifies the regularization factor in order to obtain a better convergence of matching score in depth estimation. *SmoothingCoefficient* is a positive value larger than or equal to 1.0. This value is chosen experimentally.

FileLeftViewImage

Sring, default: dog036.yuv

Specifies the filename (with the .yuv) of the original video sequence corresponds to left camera to be used for depth estimation.

FileCenterViewImage

String, default: dog038.yuv

Specifies the filename (with the .yuv) of the original video sequence corresponds to center camera to be estimated depth.

FileRightViewImage

String, default: dog040.yuv

Specifies the filename (with the .yuv) of the original video sequence corresponds to right camera to be used for depth estimation.

FileOutputDepthMapImage

String, default: depth_dog038.yuv

Specifies the filename (with the .yuv) of the output depth map video sequence.

FileCameraParameter

String, default: cam_param_dog.txt

Specifies the filename (with the .txt) of the file which includes intrinsic and extrinsic parameters of all cameras at real viewpoints.

BaselineBasis

Unsigned Int (0-3), default: 1

Specifies the basis of baseline for disparity. 0 means the minimum baseline, 1 means the maximum baseline, 2 means the baseline between center and left camera, and 3 means the baseline between center and right camera.

Precision

Unsigned Int (1 or 2 or 4), default: 4

Specifies the level of horizontal precision to find correspondances. 1 means interger-pixel precision, 2 means half-pixel precision, and 4 means quarter-pixel precision.

Vertical Precision

Unsigned Int (1 or 2 or 4), default: 4

Specifies the level of vertical precision to find correspondances. 1 means interger-pixel precision, 2 means half-pixel precision, and 4 means quarter-pixel precision.

SearchLevel

Unsigned Int (1 or 2 or 4), default: 2

Specifies the precision of candidate disparites. 1 means interger-pixel precision, 2 means half-pixel precision, and 4 means quarter-pixel precision.

Filter

Unsigned int (0-2), default: 1

Specifies the horizontal upsampling filter to generate image signals on sub-pixel positions. 0 means (bi-) linear filter, 1 means (bi-) cubic filter, and 2 means filter which used in MPEG-4 AVC.

VerticalFilter

Unsigned int (0-2), default: 1

Specifies the vertical upsampling filter to generate image signals on sub-pixel positions. 0 means (bi-) linear filter, 1 means (bi-) cubic filter, and 2 means filter which used in MPEG-4 AVC.

MatchingMethod

Unsigned Int (0-4), default: 4

Specifies the method how to identify corresponding pixels. 0 means the conventional method where disparities are given obviously, 1 means disparity-based method where disparites are calcurated by z values, 2 means homography-based method, 3 means soft segmentation method, 4 means projection pixel coordinate to side views using camera parameters, along epipolar line.

`TemporalEnhancement (ignored when DepthEstimationMode = 2)`

Bool (0 or 1), default: 1

Specifies the temporal consistency enhancement selection. The value 1 represents the use of the temporal consistency enhancement and the value 0 represents no use.

`Threshold (ignored when DepthEstimationMode = 2)`

Double, default: 2.00

Specifies the threshold of MAD. The purpose of the thresholding using MAD is separation of the static region and the moving region. If MAD of the macroblock is smaller than the threshold, we regard that the macroblock is the static region and we apply the temporal consistency enhancement.

The followings are recommended parameters for each sequence for the modified temporal consistency enhancement.

Table 1. Recommended parameters

Sequence	Threshold	Sequence	Threshold
Poznan Street	1.00	Pantomime	1.00
Poznan Hall	1.00	Dog	2.00
Undo Dancer	1.20	Newspaper	1.50
GT Fly	1.00	Lovebird1	1.50
Champagne Tower	1.00	Lovebird2	1.50

Given the configuration files in Example 6, the encoder can be started using the call in Example 7 and Example 8 with the configuration file.

`MatchingBlock (ignored when MatchinMethod = 3)`

Unsigned Int (1 or 3), default: 1

Specifies the size of block matching. # 1...Pixel matching, 3...3x3 block matching

`Occlusion`

Unsigned Int (0 or 1), default: 1

Specifies the advanced occlusion handling when pixel corenspondences is search take place. 0 mean that metric used for depth estimation is minimum of the cost of pixel matching in left and right views. 1 means that occusion apiring in the scene under depth estimation is taken acound and metic is calculated both based on left and roght view cost when given pixel is not occluded in a side view

`SmoothingCoefficient2`

Double, default: 1.0

Specifies the additional smoothing coefficient. In segmented image, adjacent two pixels are in the same segment, smoothing coefficient is "SmoothingCoefficient". Otherwise, smoothing coefficient is the product of "SmoothingCoefficient" and "SmoothingCoefficient2".

`ImageSegmentation`

Unsigned Int (0 or 1), default: 1

Specifies the image segmentation by using color information. The value 1 represents the use of the image segmentation and the value 0 represents no use.

SegmentationMethod

Unsigned Int (1 or 2 or 3), default: 3

Specifies the image segmentation method. Method 1 uses mean shift algorithm. This method is implemented by the function cvPyrMeanShift Filtering of OpenCV. Method 2 uses phyramid segmentation. This method is implemented by the function cvPyrSegmentation of OpenCV. Method 3 uses K mean clustering. This function is implemented by the function cvKMeans2 of OpenCV.

MaxCluster (Valid for only segmentation method 3)

Unsigned Int, default: 32

Specifies the maximum number of clusters for segmentation method 3. Positive integer value.

SoftDistanceCoeff (only for MatchingMethod = 3)

Double, default: 10.0

Specifies the parameter of distance coefficient in soft segment

SoftColorCoeff (only for MatchingMethod = 3)

Double, default: 20.0

Specifies the parameter of color similarity of pixels in soft segment

SoftBlockWidth (only for MatchingMethod = 3)

Unsigned Int, default: 11.0

Specifies the maximum width of soft segment

SoftBlockHeight (only for MatchingMethod = 3)

Unsigned Int, default: 11.0

Specifies the maximum height of soft segment

DepthEstimationMode

Unsigned Int (1 or 2 or 3), default: 0

0 = Automatic Depth Estimation, 1 = Semi-automatic Depth Estimation mode1 (M16391 document), 2 = Semi-automatic Depth Estimation mode2 as in (M16411 document), 3 = Reference Depth mode.

FileCenterManual (only for DepthEstimationMode = 1 and 2)

String, default: ..\\semi_automatic_input\\champagne_41

In semi-automatic depth estimation mode, manually edited depth refresh frames can be supplied via separate files. These manual input files must follow following filename-convention:
{FileCenterManual} prefix followed by MDM000.png = Manual Disparity Map or Manual Depth Map
{FileCenterManual} prefix followed by MEM000.png = Manual Edge Map
{FileCenterManual} prefix followed by MSM000.png = Manual Static Map
where 000 indicates frame 0, etc. Note that the first frame of a sequence is counted as frame 0, not 1.

When DepthEstimationMode=1 or 2, DERS will check for Manual Input files at the start of each frame. If for example following files exist:
..\\semi_automatic_input\\champagne_41MDM081.png
..\\semi_automatic_input\\champagne_41MEM081.png
..\\semi_automatic_input\\champagne_41MSM081.png
DERS will read these files at frame 81 and generate a "depth refresh frame" based on the inputted Manual Disparity Map, Manual Edge Map and Manual Static Map (see document M16391)
This enables the user to create manual input files for random frames (e.g. frame 0, 81, 83, and 100 etc)

A "depth refresh frame" can be inputted in two ways:

1.) **By providing a MDM (Manual Disparity Map), and MEM (Manual Edge Map) file for the particular framenumber**
In this case the MDM file is a Manaual Disparity Map (rough disparity map). Note that in this case gray values in MDM file represent DISPARITY values (so values from the DEPTH map can NOT be used here, you have to determine the disparity values)
DERS will create a "depth refresh frame" based on the provided MDM (Manual Disparity Map and MEM (Manual Edge Map) files.

2.) **By providing a MDM (and no MEM file) for the particular framenumber**.
In this case the MDM file must be a (manually edited) DEPTH map, and DERS uses it as a "depth refresh frame" directly. Note that in this case gray values in MDM file represent DEPTH values (not disparity values).

For example if files xxMDM000.png and xxMEM000.png exist then frame 0 is a depth refresh frame calculated from the Disparity map and Edge map. If a file cam7MDM000.png exists then the supplied image is output as depth map for frame 0.

The optional MSM file indicates which pixels have static disparity in time (until the next MSM file is found). This allows static foreground objects to be fixed static over time. If this is not required, either don't supply any MSM file, or leave all pixels in the MSM image black (gray value 0). If a pixel in the MSM file is white (or actually not zero), then the pixel is forced static and depth from the last Depth Refresh Frame is output for that pixel.

The Manual Disparity Map is used to manually set the disparity for areas where automatic depth estimation can not determine the disparity correctly (e.g. areas with low texture etc.). Note that in this case you must supply a corresponding manual edge map (MEM file).

DepthEstimationMode1 and DepthEstimationMode2 (DepthEstimationMode=1, 2 respectively) are different in the way depth is propagated from a Depth Refresh frame to following frames. Namely:

- DepthEstimationMode1 uses the same temporal consistency mechanism as automatic depth estimation (`TemporalEnhancement` is forced On automatically if DepthEstimationMode = 1). Note that parameter `Threshold` (the Mean Absolute Difference Threshold) must be set correctly for best depth estimation results.
- DepthEstimationMode2 is configured using `ThresholdOfDepthDifference`, `MovingObjectsBSize`, and `MotionSearchBSize` (see below)

`ThresholdOfDepthDifference (only for DepthEstimationMode = 2)`

Specifies the threshold of MAD of blocks in previous frame and current frame.

`MovingObjectsBSize (only for DepthEstimationMode = 2)`

Available Values: 0: small 1: medium 2: large

Specifies the block size to search a motion vector in search range

`MotionSearchBSize (only for DepthEstimationMode = 2)`

Available Values: 0: narrow 1: medium 2: wide

Specifies search range of block.

`RefDepthFile (only for DepthEstimationMode = 3)`

Specifies the path and filename of a reference depth video. When DeptEstimationMode=3, a previously generated depth map video can be input as reference. For example the (manually edited) depth video for camera 6 can be used as reference depth when generating depth for camera 8 and camera 10.

Please note that the configuration for both cameras needs to be the same, as the Znear and Zfar values and depth scaling is assumed to be the same for the reference depth and depth to be generated.

E.g. in this example the minimum and maximum Disparity Range and Disparity Search Range of camera 6,8, and 10 must be the same. Furthermore, if depth of camera 6 is one baseline (leftcam = 5 and rightcam =7) then the same distance should be used for camera 8 (left camera = 7 and right camera = 9) and camera 10 (left camera = 9, right camera 11).

`RefDepthCameraName (only for DepthEstimationMode = 3)`

Specifies the name of the camera to be used as reference depth. The name is as specified in the camara parameters files.

Example 7: Depth Estimation call `when compiled` with Visual Studio 2012

```
>DERSVC11.exe DepthEstimation.cfg
```

Example 8: Example Depth Esimationt call `when compiled` with Visual Studio 2010

```
>DERSVC10.exe DepthEstimation.cfg
```

2.3 Camera parameter file

All available configuration file parameters for the depth estimation together with a brief description are summarized in Example 6. Additional information about the camera parameters is given below.

Specification of Camera Parameters

Camera parameters shall be specified as rotation matrix **R**, translation vector **t** and intrinsic matrix **A** for each camera i. Values shall be given in floating point precision as accurate as possible.

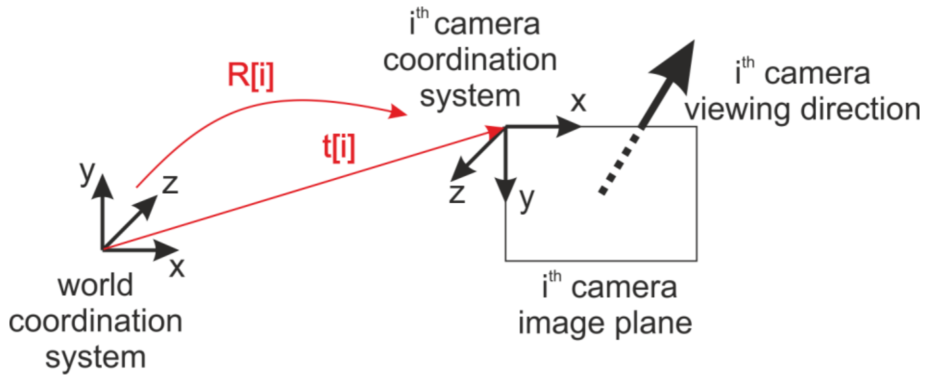
i^th camera coordination system
R[i]
t[i]
i^th camera viewing direction
x
z
y
y
z
x
world coordination system
i^th camera image plane

Figure 2: Relation between i^th camera coordination system and world coordination system

The extrinsic camera parameters **R** and **t** shall be specified according to a right-handed coordinate system. The rotation matrix **R**[i] for i-th camera is represented as follows.

r_11[i]	**r_12**[i]	**r_13**[i]
r_21[i]	**r_22**[i]	**r_23**[i]
r_31[i]	**r_32**[i]	**r_33**[i]

The translation vector **t**[i] for i-th camera is represented as follows:

t_1[i]
t_2[i]
t_3[i]

The rotation matrix **R** and the translation vector **t** define the position and orientation of the corresponding camera with respect to the world coordinate system (fig 2). The components of the rotation matrix **R** are function of the rotations about the three coordinate axes.

The upper left corner of an image shall be the origin for corresponding image/camera coordinates, i.e., the (0,0) coordinate, with all other corners of the image having non-negative coordinates.

The intrinsic matrix **A**[i] for i-th camera is represented as follows:

focal_length_x[i]	**radial_distortion**[i]	**principal_point_x**[i]
0.0	**focal_length_y**[i]	**principal_point_y**[i]
0.0	0.0	1.0

focal_length_x[i] specifies the focal length of the i-th camera in the horizontal direction units of pixels.

focal_length_y[i] specifies the focal length of the i-th camera in the vertical direction in units of pixels.

principal_point_x[i] specifies the principal point of the i-th camera in the horizontal direction units of pixels.

principal_point_y[i] specifies the principal point of the i-th camera in the vertical direction in units of pixels.

radial_distortion[i] specifies the radial distortion coefficient of the i-th camera.

- Radial distortion coefficients are estimated up to first order
- Radial distortion centre is the same as the principal point.
- Tangential distortion coefficients have not been estimated, they are to be assumed zero.

Example 9: Example camera parameter file

```
'camera name of the 0th view'
focal_length_x[0] radial_distortion[0] principal_point_x[0]
0.0               focal_length_y[0]    principal_point_y[0]
0.0               0.0                  1.0
0.0
0.0
r_11[0]           r_12[0]              r_13[0]         t_1[0]
r_21[0]           r_22[0]              r_23[0]         t_2[0]
r_31[0]           r_32[0]              r_33[0]         t_3[0]

'camera name of the 1sr view'
focal_length_x[1] radial_distortion[1] principal_point_x[1]
0.0               focal_length_y[1]    principal_point_y[1]
0.0               0.0                  1.0
0.0
0.0
r_11[1]           r_12[1]              r_13[1]         t_1[1]
r_21[1]           r_22[1]              r_23[1]         t_2[1]
r_31[1]           r_32[1]              r_33[1]         t_3[1]

'camera name of the 2nd view'
focal_length_x[2] radial_distortion[2] principal_point_x[2]
0.0               focal_length_y[2]    principal_point_y[2]
0.0               0.0                  1.0
0.0
0.0
r_11[2]           r_12[2]              r_13[2]         t_1[2]
r_21[2]           r_22[2]              r_23[2]         t_2[2]
r_31[2]           r_32[2]              r_33[2]         t_3[2]
```

Example 10: Example camera parameter file "cam_param_poznan_blocks.txt"

```
param_cam0
1731.8537611304296    0.0              944.82280521513485
   0.0            1730.2451743420156  522.20735217757431
   0.0                0.0                1.0
0
0
 0.466974 -0.484961  0.739424 -33.50130
-0.763243 -0.643311  0.060092  -4.40250
 0.446537 -0.592422 -0.670552  -6.01074

param_cam1
1728.8073564366866    0.0              967.09929177657295
   0.0            1727.1835232972953  500.79439880155053
   0.0                0.0                1.0
0
0
 0.575101 -0.535786   0.618217 -30.70100
-0.735873 -0.668962   0.104786  -7.27099
 0.357421 -0.515192  -0.778991  -2.13728

param_cam2
1726.6398135686711    0.0              952.00110018182022
   0.0            1725.1218369974608  504.02369575109100
   0.0                0.0                1.0
0
0
 0.652131 -0.607829  0.453067 -27.055300
-0.717793 -0.687346  0.111035 -10.648300
 0.243923 -0.397617 -0.884535   0.856718

param_cam3
1723.8134067272049    0.0              936.34465010886481
   0.0            1719.2642370302140  551.01088851627435
   0.0                0.0                1.0
0
0
 0.718809 -0.639157  0.273482 -23.14950
-0.687062 -0.713153  0.139132 -14.45100
 0.106107 -0.287909 -0.951761   2.86124

param_cam4
1721.1073359005352    0.0              936.36509770885857
   0.0            1716.9571493450005  546.95869231404413
   0.0                0.0                1.0
0
0
 0.7595750 -0.643865  0.0921044 -18.90780
-0.6495260 -0.743465  0.1592980 -18.05480
-0.0340901 -0.180823 -0.9829250   4.16231
```

3 Information for Software Integration

At the MPEG meetings, the integration of proposals into the DERS is decided. The software coordinators collect information on the integration complexity of all proposals and arrange a schedule for the integration period after the meeting. As input for this procedure, proponents should prepare to give an estimate of the time required for the integration of their proposal by the end of the meeting. The integration time includes the integration work itself and validation testing by the proponent.

The following software integration process is proposed:

- Please check out the latest software version from the MPEG SVN.
- In case you might be running into trouble meeting the deadline please contact the software coordinators as soon as possible.
- Propose new tests that are designed to ensure that the tool you have implemented is not broken. Therefore, the tests do not need to demonstrate the performance of the tool but should rather validate its operability.
- Update the file "*ders_changes.txt"* as well as the software manual.

To verify the validity of the claimed gains of the proposal, proponents are encouraged to provide verification results with the integrated DERS for their adopted tool at the next meeting.

3.1 Software integration guidelines and rules

When integrating adopted proposals into the DERS, please obey the following basic principles and recommendations:

- **The integrated software shall compile without warnings when using the provided VC11 and VS10 solutions as well as linux makefiles (i.e. using an up-to-date g++ compiler).**
- Follow the coding style of the Depth Estimation software. Use 2 (two) spaces for indentation, no tabs.
- Re-use code and integrate functionality as possible. Try to avoid redundant code.
- Do not change the meaning of existing input parameters but define new ones if necessary (and applicable).
- Make sure that new parameters have meaningful default values. Tools should not be switched on by default (if not decided different by the MPEG).
- Do not re-structure the output of the compiled binaries (if not decided different by the MPEG).
- Please change the DERS version number VERSION located in the file "version.h" inline with your integration tag (e.g. "6.1").

3.2 Information to be provided to the software coordinators

The following information shall be provided after integration of a proposal into the DERS.

- The corresponding proposal numbers (for reference),
- A zipped software package containing only the modified files, but in the correct directory structure *without SVN directories*,
- The output for the current validation scripts when running with the modified software,
- Proposals for modified and/or additional test algorithms for existing tools.
- Proposals for new short term tests (one or more) that validate the functionality of the new tool.
- A brief list of changes (to maintain the changes file "*ders_changes.txt*" in the root directory of the DERS – the changes should be directly written to the file "*ders_changes.txt*"),
- An updated version of the software manual (when parameters or tools have been added) with changes marked so that they can be easily located and identified.