Friedrich-Alexander-Universität Erlangen-Nürnberg

Multimedia Communications and Signal Processing

Prof. Dr.-Ing. André Kaup

Research Internship

HAM - H.265/HEVC Analyzer for MATLAB

Andreas Luder

April 2015

Supervisors: Michel Bätz, Andrea Eichenseer
Contents

List of Abbreviations I

1 Introduction 1

2 Side information on H.265/HEVC 3
  2.1 Block organization in H.265/HEVC . . . . . . . . . . . . . . . . . . . . 3
  2.2 Motion Estimation . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
  2.3 Quantization . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
  2.4 Intra Coding . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

3 HM module of HAM 7
  3.1 System requirements . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
  3.2 Installation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
  3.3 How the HM module works . . . . . . . . . . . . . . . . . . . . . . . . 9

4 HAM MATLAB interface 13
  4.1 HAM MATLAB GUI . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
  4.2 HAM Export . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20

References 23
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAM</td>
<td>H.265/HEVC Analyzer for MATLAB</td>
</tr>
<tr>
<td>HARP</td>
<td>H.265/HEVC Analyzer for Rapid Prototyping</td>
</tr>
<tr>
<td>HM</td>
<td>H.265/HEVC Test Model</td>
</tr>
<tr>
<td>GOP</td>
<td>Group Of Pictures</td>
</tr>
<tr>
<td>QP</td>
<td>Quantization Parameter</td>
</tr>
<tr>
<td>CTU</td>
<td>Coding Tree Unit</td>
</tr>
<tr>
<td>CU</td>
<td>Coding Unit</td>
</tr>
<tr>
<td>PU</td>
<td>Prediction Unit</td>
</tr>
<tr>
<td>PB</td>
<td>Prediction Block</td>
</tr>
<tr>
<td>TU</td>
<td>Transform Unit</td>
</tr>
<tr>
<td>POC</td>
<td>Picture Order Count</td>
</tr>
<tr>
<td>DST</td>
<td>Discrete Sine Transform</td>
</tr>
</tbody>
</table>
The goal of this internship is to modify the chair’s existing H.265/HEVC Analyzer for Rapid Prototyping (HARP) toolkit in a way that it is able to export coding information of single frames in a form which is readable by MATLAB [LMS15].

Even though HAM was developed as an add-on to HARP, the final version is independent from HARP and can be used stand-alone. However, the stand-alone version needs the HEVC Test Model (HM) to be installed on the system.

As mentioned before, the main goal of HAM is to export data which is created during the encoding of a video sequence in a way that it is readable by third party software. As addition to this feature, HAM also provides a graphical user interface (GUI) for MATLAB which visualizes the previously exported data.
Chapter 2

Side information on H.265/HEVC

This chapter provides information on some of the features of the H.265/HEVC coding standard that might be useful to understand the different options and capabilities of HAM. Please note that basic knowledge on the H.265/HEVC coding standard is assumed in this tutorial. Basic knowledge can be obtained through [SOHW12]. Detailed information on the standard is available online [IEE12].

2.1 Block organization in H.265/HEVC

In H.265/HEVC coding blocks are organized in a quad-tree based fashion with the Coding Tree Unit (CTU) as the top structure. The maximum size of a CTU, which can be set in the encoder, is 64 times 64 pixel. For prediction, a CTU can be split into smaller Prediction Units (PU) by the encoder. H.265/HEVC allows symmetric and non-symmetric PUs in various sizes. The maximum size of a PU is 64 times 64 pixel which is the maximum size of a CTU. For Transformation, a CTU is split into smaller Transform Units (TU). In the case of Transform Units, only symmetrical blocks are allowed with a maximum size of 32 times 32 pixel. Figure 2.1 illustrates the different splitting options of PUs [SOHW12].
2.2 Motion estimation

Similar to its predecessors, H.265/HEVC is able to perform P-prediction as well as B-prediction. In the case of P-prediction, the encoder selects reference pictures which have a lower POC than the current picture and keeps them in a reference picture list which is used for predicting the current frame. If B-prediction is performed, the encoder also uses frames with a higher POC for predicting the current frame. In the case of B-prediction, the encoder uses two reference picture lists. Each reference picture list contains long-term and short-term reference pictures [SCF+12]. Since previous and future reference pictures are used in B-prediction, a B-predicted block has two motion vectors.

2.3 Quantization

In H.265/HEVC, the granularity of the quantization is defined through a quantization parameter (QP). The value of this QP has to be selected by the user in the range from 0 to 51. However, this user-given value for QP serves only as a QP base value. This base value is altered by the configuration profile of H.265/HEVC. Hence, each frame within the Group Of Pictures (GOP) can have a different value for the base QP of each
frame. Additionally, H.265/HEVC offers options for dynamic QP selection depending on the content of the input sequence. One option enables the encoder to select the best fitting QP value in the interval \([-\text{range}, \text{range}]\) at slice level. In HM, the parameter \(-\text{dqr}<\text{range}\) enables this option. Another option for dynamic QP selection effects the QP value at Coding Unit (CU) level. With this option enabled, the encoder searches at CU level for the best fitting QP value in the interval \([-\text{range}, \text{range}]\). In HM, the parameters \(-\text{MaxCuDQPDepth}=<\text{maxDepth}\) and \(-\text{d}<\text{range}\) enable this option.

There is also an option for adaptive QP selection, however, influences of this option in the resulting data could not be measured during testing. One reason for this behavior might be, that for our test sequences the adaptive QP selection chose similar QP values as the non-adaptive configuration.

2.4 Intra Coding

In H.265/HEVC, there are 35 different modes for intra coding of luma CUs available. These are DC, planar and 33 directional modes. Since the chroma component is often correlated with the luma component, the chroma coding block has only five different modes \([\text{LBH}^{+12}]\). Those are DC mode, planar mode, horizontal mode, vertical mode, and derived mode (DM). Derived mode means that the chroma component of an intra coded CU has the same intra mode as the luma component. If the encoder selects either DC, planar, horizontal, or vertical mode for an intra coded luma Prediction Block (PB), intra angular mode 34 can be selected as a fifth option for the according intra coded chroma PB by the encoder. This behavior was implemented in order to remove possible redundancy in the signaling \([\text{LBH}^{+12}]\). Fig. 2.2 shows the different angular modes for intra predicted luma blocks.
Figure 2.2: Intra angular modes for luma prediction blocks [LBH+12]
Chapter 3

HM module of HAM

This chapter provides information on the HM module of HAM. First, the system requirements will be laid out, in the second part, the installation of the HM module will be depicted step by step. Finally, it will be explained how the HM module works.

Overall, HAM consists of four files. However, the HM module of HAM was implemented only in one single file, “HAM.h”. In the other files, the MATLAB functionality of HAM is realized.

3.1 System requirements

HAM requires HM and OpenCV to be installed on the system [OP315]. HAM was tested with the latest version of HM (version 16.3 at this time). The installed OpenCV version during testing was 2.4.9. Due to the vast changes which come with every update of HM, it is possible that HAM needs to be adapted in order to work with other versions of HM. The installation of HM and OpenCV is not subject to this tutorial as there are already excellent guides available online.

3.2 Installation

To successfully install HAM, the following steps have to be conducted:
• Copy HAM.h into /HM_DIRECTORY/source/Lib/TLibEncoder/

• In the include section of TEncSlice.cpp
  (/HM_DIRECTORY/source/Lib/TLibEncoder/TEncSlice.cpp) add the line:
  
  # include “HAM.h”;

• In TEncSlice.cpp (/HM_DIRECTORY/source/Lib/TLibEncoder/TEncSlice.cpp),
  at the end of the function “void TEncSlice::compressSlice(TComPic* pcPic)”
  (approximately line 920) add:
  
  HAM(pcPic, boundingCtuTsAddr);

• In HAM.h change the path to which HAM exports the information of encoded
  frames. In line 117 change variable “exportPath” accordingly. Please make sure
  that a valid directory is set.

• In file /HM_DIRECTORY/build/linux/lib/TLibEncoder/makefile, change lines
  16 and 17 into:
  
  USER_INC_DIRS = -I$(SRC_DIR) -I/HOMES/USERNAME/include
  USER_LIB_DIRS = -L/HOMES/USERNAME/lib

• In file /HM_DIRECTORY/build/linux/lib/TLibEncoder/makefile, below
  “# defines to set”, add:
  
  DEFS = -std=c++11
  DEFS = -w

• In file /HM_DIRECTORY/build/linux/app/TAppEncoder/makefile, change lines
  15 and 16 into:
  
  USER_INC_DIRS = -I$(SRC_DIR) -I/HOMES/USERNAME/include
  USER_LIB_DIRS = -L/HOMES/USERNAME/lib

• In file /HM_DIRECTORY/build/linux/app/TAppEncoder/makefile, below
  “# defines to set”, add:
DEFS = -std=c++11
DEFS = -w

- In file /HM_DIRECTORY/build/linux/app/TAppEncoder/makefile, change
  LIBS = -ldl (line 41) into:
  LIBS = -ldl -lopencv_core -lopencv_imgproc -lopencv_features2d
  -lopencv_highgui

- Enjoy HAM

### 3.3 How the HM module works

The task of the HM module of HAM is to export all kinds of useful information into a format that is readable by MATLAB. During the process of encoding a video sequence, the HAM HM module writes three files for each frame. For example, for frame 1 of a video sequence, the files POC00001.Orig.jpg, POC00001.PredictionUnits.txt, and POC00001.TransformUnits.txt are written. The JPG file contains an image of the source frame, the text files contain information about every prediction unit/transform unit of the frame. In the text files, each line represents one unit. An additional file, GeneralInformation.txt, which contains general information about the video sequence is also written. The following information is stored for each Prediction Unit:

- Starting position in x direction
- Starting position in y direction
- Block dimension in x direction
- Block dimension in y direction
- Block type (intra=0, skip=1, merge=2, inter=3)
• X component of motion vector with reference from list 0
• Y component of motion vector with reference from list 0
• X component of motion vector with reference from list 1
• Y component of motion vector with reference from list 1
• Base QP value for current slice
• QP value of the current coding unit
• Slice type (I=0, P=1, B=2)
• Chosen reference list (list0=0, list1=1, both lists=10)
• Distance of current POC to reference picture of list 0
• Distance of current POC to reference picture of list 1
• Intra luma mode (DC=0, planar=1, intra angular modes=2-34)
• Intra chroma mode (DC=0, planar=1, horizontal mode=10, vertical mode=26, intra angular mode 34=34, derived mode=36)
• Starting position in x direction of the attached CTU
• Starting position in y direction of the attached CTU
• CTU index of the attached CTU

The following information is stored for each Transform Unit:

• Starting position in x direction
• Starting position in y direction
• Block dimension in x direction
• Block dimension in y direction
• Block type (intra=0, inter=1, TUs with residual not coded=2, skip=3, intra blocks of size 4x4 (DST coded blocks)=4)

Internally, the HM module creates a struct for each CTU in the current frame. This struct contains the starting position, the dimension and the index of the CTU. Each struct also contains two lists which contain all prediction units/transform units that belong to this CTU. After one frame has finished processing, all information about this frame is written to the according files. For more information on how the code works, please refer to the comments in the code.
Chapter 4

HAM MATLAB interface

This chapter provides information on the MATLAB module of HAM. The functionality of the HAM MATLAB GUI as well as each mode will be explained in detail. Afterwards, the functionality of the automatic export function will be laid out.

The MATLAB GUI consists of the files “HAM_GUI.fig” and “HAM_GUI.m”. The function to automatically export information for multiple frames is realized in the file “HAMexport.m”.

4.1 HAM MATLAB GUI

HAM provides a MATLAB GUI which is able to display information that was previously exported from HM with the HAM HM module. The MATLAB GUI is capable of saving all information of a frame in a user-friendly way. For quick access to relevant data for a frame, the data of said frame is saved in different matrices. Figure 4.1 shows the HAM MATLAB GUI after startup. This startup view depends on the user’s MATLAB environment and might look different with various settings of the MATLAB workspace, i.e., some areas might be blank.

In Figure 4.1, the different areas of the HAM MATLAB GUI are marked in different colors. Please note that there are no colored boxes in the real GUI. The blue marked
Figure 4.1: HAM GUI startup window

area graphically displays the data of the selected frame. In the top left corner, the original frame is displayed. In the top right corner, a close-up of a specific CTU is displayed. The CTU which is displayed close-up is also highlighted in the original frame. The bottom image displays the whole frame with overlaying coding information. The red marked area shows the data of a frame more accurately in histograms. In the green marked area, settings concerning the video sequence, frame number and the CTU to be viewed close-up can be selected. One can either use the sliders or alternatively type the desired frame or CTU number into the according text box. The “Change folder” button opens a dialog in which the working directory of HAM can be changed. A click on the “Export data” button saves all relevant data of the current frame to the file “Frame_FRAME_NUMBER_data.mat” into the working directory. The current working directory of HAM can also be seen in the green marked area.

The control elements marked in yellow change the data that is displayed in the image area as well as the diagram area. In the “Controls” panel, the user can choose which type of information is to be displayed. The different options contain prediction units,
transform units, motion vectors, reference indices, luma mode, chroma mode, and QP map.

The “Modes” panel changes its content depending on the selected option in the “Controls” panel. In the “Modes” panel, the displayed data can be selected and un-selected. In the purple marked area, information on the axis of the histograms can be found.

The starting behavior of HAM varies depending on some environment variables. During the launch of HAM, the current MATLAB directory is set as working directory

**Show prediction units**

In the “Show prediction units” mode, the different prediction units of a frame and the partitioning are displayed. H.265/HEVC has four different types of prediction units. Inter (displayed in blue), Inter-Merge (displayed in seagreen), Inter-Skip (displayed in green) and Intra (displayed in red). The output image and the close-up CTU are colored accordingly. The upper histogram (histogram 1) shows the distribution of pixels of the whole frame into the different prediction unit types in percent. The lower histogram (histogram 2) shows the distribution of different block sizes in absolute numbers. The x-axis of this diagram is labeled from 0 to 25. The partitioning of the block sizes can be seen through the “Histogram info” popup list which can be found below the “Modes” panel. In the “Modes” panel, the different types of prediction units can be shown/hidden. The images as well as histogram 2 adapt accordingly. Figure 4.2 shows the appearance of HAM in the “Show prediction units” mode.

**Show transform units**

Similar to the “Show prediction units” mode, the “Show transform units” mode shows the partitioning of the transform units of a frame. H.265/HEVC only allows quadratic transform blocks. Histogram 2 has been modified to only display quadratic blocks. For intra transform blocks of size 4 times 4 a Discrete Sine Transform (DST) is applied. Histogram 1 also shows a bar for DST pixels. Figure 4.3 shows the appearance of HAM in the “Show transform units” mode.
Figure 4.2: Show prediction units

**Show motion vectors**

In the “Show motion vectors” mode, the motion vectors for each coding unit are drawn into the image. H.265/HEVC uses P-prediction and B-prediction. P-prediction uses a reference frame from one reference picture list and bi-prediction uses one reference frames from each of the two reference picture lists. Therefore, two different colors are used for the motion vectors. Motion vectors with reference from reference picture list 0 are colored magenta, while vectors from list 1 are colored yellow. Zero motion vectors are displayed either as a magenta + or as a yellow x. The appearance and behavior of all other controls and diagrams are similar to the “Show prediction units” mode. The appearance of HAM in 'Show motion vectors' mode can be seen in Figure 4.4.

**Show reference indices**

In the default mode of “Show reference indices”, P-predicted coding units (CUs) with reference in reference picture list 0 are displayed in yellow color. P-predicted CUs with reference in reference picture list 1 are displayed in blue color. Both colors get
4.1. HAM MATLAB GUI

Figure 4.3: Show transform units
darker the further away the picture order count (POC) of the reference is from the current POC. B-predicted CUs are displayed yellowish with blueish dots. Histogram 1 depicts the number of blocks which have references only from list 0, only from list 1, or from both lists. Histogram 2 shows the relative distance of the reference block POC to the current POC. For example, when the current POC is 10 and the relative reference POC is -2, the absolute POC of the reference is 8. In histogram 2, references from list 0 are displayed in ultramarine, references from list 1 are displayed in cyan. Bi-predicted CUs with references from both lists have yellow for list 0 and red for list 1. Once again, the “Modes” panel offers options to show/hide different types of blocks. In this panel, there is also the option to switch to alternative displaying method of the “Show reference indices” mode. In this mode, the doted blocks are replaced with a block that consists of two triangles. This method of displaying might be useful if distances of POCs are to be investigated more easily. Please note that in the “Show reference indices” intra coded CUs are transparent. Figures 4.5 and 4.6 show the two appearances of this mode of HAM.
Show intra luma

In the “Show intra luma” mode, intra blocks are colored depending on their intra mode. Histogram 1 shows the number blocks according to their sizes. Histogram 2 illustrates the amount of blocks with different colors according to their intra mode. Histogram 2 also changes depending on the options selected in the “Modes” panel. In the image, different colors for DC mode (light blue), planar mode (white), horizontal mode (red), and vertical mode (green) are applied. These modes have separate options to be shown/hidden in the ’Modes’ panel. Intra angular modes 2-3 (ultramarine), intra angular modes 4-9 (violet), intra angular modes 11-25 (yellow), intra angular modes 27-32 (gras green), and intra angular modes 33-34 (dark green) can only be shown/hidden at once. In histogram 2, DC mode is represented in ultramarine, planar mode in light blue, horizontal mode in light green, vertical mode in orange, and all other modes in red. Figure 4.7 shows the appearance of HAM in the “Show intra luma” mode.
Show intra chroma

The “Show intra chroma” mode behaves quite similar to the “Show intra luma” mode, except that all five possible chroma modes can be shown/hidden separately. In the images, DC mode is colored light blue, planar mode is colored white, horizontal mode is colored red, vertical mode is colored green, derived mode is colored yellow, and intra angular mode 34 is colored dark green. In histogram 2, the contributions of DC mode (ultramarine), planar mode (light blue), horizontal mode (turquoise), vertical mode (yellow), intra angular mode 34 (orange), and derived mode (red) are shown/hidden according to the related settings in the “Modes” panel. Figure 4.8 shows the appearance of HAM in the “Show intra chroma” mode.

Show qp map

In this mode, the different QP values are visualized. The QP which is equal to the base QP is colored yellow. QP values smaller than the base QP are colored in shades of green, QP values greater than the base QP are colored in shades of orange. Histogram
Figure 4.6: Show reference indices 2

1 shows the number of CUs smaller, equal, or greater to the base QP respectively. Histogram 2 is a more detailed version of histogram 1. In histogram 2, there are different bars for every QP value used in the current frame. The bar for QP equal to the base QP is plotted in light green, bars for QPs smaller than the base QP are plotted in ultramarine and bars for QPs greater than the base QP are plotted in red. The “Modes” panel offers options to show/hide different groups of QP values in the images as well as in histogram 2. Figure 4.9 shows the appearance of HAM in the “Show qp map” mode.

### 4.2 HAM Export

HAM Export is a MATLAB function that has the same functionality as the “Export” button in the HAM GUI. With this function, one can easily export data of many frames at once and automatically. HAM Export saves the data of all frames in a given directory in a user-friendly way. All relevant data is saved in different matrices to
4.2. HAM EXPORT

easily access data of single features of HAM. The function is called with a string as parameter which contains the path to the directory to be exported. For each frame in that directory, a file named “Frame_ #_data.mat” is created in the same directory.

Figure 4.7: Show intra luma
Figure 4.8: Show intra chroma

Figure 4.9: Show qp map
Bibliography


