Complexity reduction for intra mode selection in HEVC using OpenMP

Under guidance of Dr.K.R.Rao

Submitted By,
Pratik D Mehta
M.S.E.E
ID # 1000859338
**Acronyms**

**API**: Application Programming Interface

**AVC**: Advanced Video Coding

**CABAC**: Context Adaptive Binary Arithmetic Coding

**CB**: Coding Block

**CPU**: Central Processing Unit

**CTB**: Coding Tree Block

**CTU**: Coding Tree Unit

**CU**: Coding Unit

**CUDA**: Compute Unified Device Architecture

**DCC**: Data Compression Conference

**DCT**: Discrete Cosine Transform

**DST**: Discrete Sine Transform

**HEVC**: High Efficiency Video Coding

**ISO**: International Organization for Standardization

**ITU-T**: International Telecommunication Union – Telecommunication Standardization Sector

**JCT-VC**: Joint Collaborative Team on Video Coding

**MC**: Motion Compensation

**MCP**: Motion Compensated Predication

**OPENMP**: Open Multiprocessing

**PB**: Prediction Block

**PCM**: Pulse Code Modulation

**PU**: Prediction Unit
**SAO:** Sample Adaptive Offset

**SIMD:** Single Instruction Multiple Data

**TB:** Transform Block

**VCIP:** Visual Communication and Image Processing Conference
Proposal:

To reduce encoding time of intra mode selection in HEVC using parallel processing techniques.

Overview of HEVC:

In April 2010 a Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T ISO/IEC started its work on a new standard for High Efficiency Video Coding (HEVC). HEVC has been designed to address essentially all existing applications of H.264 [25] and to particularly focus on issues such as increased use of parallel processing techniques and increased video resolution. Three profiles, namely main, main intra and main 10 bit profiles have been finalized as final draft international standard (FDIS) by JCT-VC in Jan.2013. Apart from that various extensions such as 3D video, scalable video coding (SVC), etc are under development. While the highest performance gain also comes with associated high complexity requirements, just marginally lowering performing solutions also brings high coding gains [9][4][6]. Coding gains in HEVC are due to both advanced inter and intra predictions.

HEVC [1][11][16][17] implements the same hybrid approach as H.264 which includes both temporal and spatial predictions. It aims at 50% compression over H.264 while maintaining similar video quality [5]. It requires half the bandwidth as compared to H.264 for high quality video transmission. H.264 divides the transmission image into 16 by 16 pixels while HEVC divides the image into varying block sizes up to 64 by 64 pixels. This larger block size typically enables better compression. Various features are introduced in the HEVC standard to enhance parallel processing capability or modify the structuring of slice data for packetization purposes [2]. Either the image is divided into various parts like tiles in which processor works on one of them, or the wave front method where each processor handles one block of line in the image, or make use of dependent slice method in which dependent slice can only be decoded if part of previous slice has been decoded.

Unlike H.264 which contains 16 by 16 size micro blocks, HEVC employs quad tree structure which contains coding tree unit (CTU), size of which is selected by the encoder and can be larger than traditional micro block.
Figure 1: Division of an image into CTU [13]

Figure 1 shows the basic division of image into multiple CTUs. The width and height of CTU are signaled in a sequence parameter set hence all the CTUs in a video sequence have the same size i.e. 64 by 64, 32 by 32, or 16 by 16 [13] as shown in Figure 2.

Figure 2: Different sizes of CTU [13]

Each coding unit basically consists of luma and chroma prediction blocks and each block is called coding tree block (CTB) having the same size as CTU. But CTBs are too big to decide type of prediction method to be used. So CTBs are further divided into coding blocks (CB) which are the decision point where decision is taken whether to perform inter-picture or intra-picture prediction [13]. CBs are good enough for prediction type decision but too large to store motion vectors. Thus each CB can be split to prediction blocks (PB) differently depending on the temporal and/ or spatial predictability.
Figure 3: Encoder block diagram of HEVC [1]

Figure 3 shows block diagram of HEVC encoder in which each picture is partitioned into blocks of different sizes and the same is conveyed to the decoder. In the given sequence intra prediction is applied to the very first picture which uses spatial redundancy of the picture while for rest of the frames temporal redundancy is exploited using inter prediction [1].

Figure 4: Decoder block diagram of HEVC [E6]
The residual signal of intra/inter prediction which is the difference between original and predicted block is further transformed by a linear spatial transform which is scaled, quantized, entropy coded and transmitted along with prediction information. This residual signal is also inverse transformed, inverse quantized and filtered to duplicate the decoder processing loop and added with predicted signal to produce decoded picture which is stored in buffer for further predictions. As shown in Figure 4 in the block diagram of HEVC decoder, the residual signal is added to the prediction, and the result is fed to the deblocking filter to reduce the artifacts and finally stored in decoded picture buffer which can be used for further decoding of remaining pictures.

A non-linear amplitude mapping is introduced in the inter-prediction loop after the deblocking filter called sample adaptive offset (SAO). The goal is to better reconstruct the original signal amplitudes by using a look-up table that is described by a few parameters that can be described by histogram analysis at the encoder side [1].

**Intra Prediction**

In H.264, intra prediction [19][21][22][24] is based on spatial extrapolation of samples from previously decoded image blocks, followed by integer discrete cosine transform (DCT) [20] based coding [E3]. HEVC utilizes the same principle, but further extends it to efficiently representing wider range of textural and structural information in images. HEVC contains several elements improving the efficiency of intra prediction over earlier solutions. The introduced methods can model accurately different structures as well as smooth regions with gradually changing sample values.
As shown in Figure 5 the reference samples from top left and above the image block to be predicted and denoted by $R_{x,y}$ while predicted block is denoted by $P_{x,y}$ where $(x,y)$ denotes position of predicted sample value. In some cases neighboring reference samples may be unavailable for intra prediction then in such cases missing reference samples on the top boundary are obtained by copying the closest available reference sample [3]. In Figure 5 the missing reference samples on the top boundary are obtained by copying the closest available reference sample from the left while missing reference samples on the left boundary are generated by copying the reference samples below.

![Figure 5: Reference samples $R_{x,y}$ used in prediction to obtain predicted samples $P_{x,y}$ for a block of size $N$ by $N$ samples [3].](image)
HEVC introduces 33 angular prediction modes along with planar and DC prediction modes as shown in Figure 6. The number and angularity of prediction directions are selected to provide a good tradeoff between encoding complexity and coding efficiency [7]. In HEVC there are four effective intra prediction block sizes ranging from 4 by 4 to 32 by 32 samples, each of which supports 33 distinct prediction directions. In order to further simplify the process, all sample locations within one prediction block are projected to a single reference row or column depending on the directionality of the selected prediction mode for example, utilizing the left reference column for angular modes 2 to 17 and the above reference row for angular modes 18 to 34 [3].
### Table: Total Intra Angular modes

<table>
<thead>
<tr>
<th>Prediction size</th>
<th>HEVC/H.265 (64x64)</th>
<th>H.264/AVC (16x16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64x64</td>
<td>4</td>
<td>NA</td>
</tr>
<tr>
<td>32x32</td>
<td>35</td>
<td>NA</td>
</tr>
<tr>
<td>16x16</td>
<td>35</td>
<td>4</td>
</tr>
<tr>
<td>8x8</td>
<td>35</td>
<td>9</td>
</tr>
<tr>
<td>4x4</td>
<td>18</td>
<td>9</td>
</tr>
<tr>
<td>Total Modes</td>
<td>7808</td>
<td>16x(16x9+4x9+4)=2944</td>
</tr>
</tbody>
</table>

*Figure 8: Comparing HEVC Intra luma prediction modes for 64x64 LCU with H.264/AVC Intra modes for a 64x64 image region [26]*

*Figure 7 shows 9 different modes supported by H.264 while Figure 8 shows comparison of number of prediction modes supported by HEVC and H.264 corresponding to different block sizes [26].

---

**Parallel Programming**

Parallel computing is basically a technique in which multiple computations are done simultaneously. The basic approach for parallel processing is to break the task into multiple smaller tasks and further assign each task to each of the thread which performs required operations in parallel [12]. Parallelization can sometimes get complicated due to race conditions, data dependency, synchronization and communication among different threads.

The basic approach in deciding parallelization approach is to first analyze the part of the program that needs to be parallelized and then decide the type of parallel programming technique that needs to be implemented. It should also be determined whether or not the problem is one that can be parallelized [8][12][23].

Parallel programming models are not limited to particular type of machinery but can be implemented on any underlying hardware.
**OpenMP**

The OpenMP API uses the fork-join model for parallel execution in which multiple threads of execution perform tasks defined implicitly or explicitly by OpenMP directives [14] as shown in Figure 8. The OpenMP API is intended to support programs that will execute correctly both as parallel programs and as sequential programs. It supports multi-platform shared memory multiprocessing programming.

![Illustration of multithreading in OpenMP](image)

*Figure 8: Illustration of multithreading in OpenMP [12]*

The OpenMP API provides a relaxed-consistency, shared-memory model. All OpenMP threads have access to a place to store and to retrieve variable, called the memory. In addition, each thread is allowed to have its own temporary view of memory which allows the thread and the memory for every reference to a variable. Each thread also has a access to another type of memory that must not be accessed by other threads, called thread private memory [E1][E4].

**Proposed Work**

The aim of the proposed work is to reduce the encoding time required for intra prediction using parallel techniques. The intra prediction scheme in HEVC contains total of 35 different predictions as shown in Figure 6 modes as compared to 9 different modes in H.264 in Figure 7 [12][1]. This further increases complexity and encoding time of HEVC in selecting the appropriate mode [10].

Hence if a scheme is designed that will calculate all the 35 different prediction modes using task based parallelism then it will further reduce encoding time without much increase in complexity.
E1 Thesis by S.Gangavati on “Complexity reduction of H.264 using parallel programming” which describes significant speed-up in encoding time on GPU using CUDA and CPU combined than on CPU by data and task parallelization, 2012.


E4 Thesis by T.Sathe on “Complexity reduction in H.264 encoder using OpenMP” which basically makes use of parallel processing approach using threads that are managed by OpenMP, 2012.

E5 Thesis by P.K.Gajalla on “Efficient HEVC loss less coding using sample based angular intra prediction” which describes use of sample based angular prediction approach which can be used for better intra prediction accuracy compared to HEVC block angular intra prediction, 2013.


P.S The above mentioned Theses/Projects can be accessed from MPL Website of University of Texas at Arlington http://www-ee.uta.edu/Dip/Courses/EE5359/index.html.
References


https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/HM-6.0

[12] Introduction to parallel computing

[13] Information about quad tree structure of HEVC

http://bisqwit.iki.fi/story/howto/openmp/
[15] Website for downloading test sequence for research purposes
http://media.xiph.org/video/derf/

[16] Information on developments in HEVC NGVC- Next generation video coding
http://bisqwit.iki.fi/story/howto/openmp/

online available: http://phenix.int-evry.fr/jct/doc_end_user/documents/6_Torino/wg11/JCTVC-F634-v2.zip

[18] JCT-VC documents are publicly available at http://ftp3.itu.ch/av-arch/jctvc-site and
http://phenix.it-sudparis.eu/jct/

[19] T.L Silva et al, ”HEVC intra coding acceleration based on tree inter-level mode
correlation”, SPA 2013 Sep.2013, Poznan, Poland


[21] H. Zhang and Z. Ma, ”Fast intra prediction for high efficiency video coding “, Pacific

[22] M. Zhang, C. Zhao and J. Xu, ”An adaptive fast intra mode decision in HEVC “, IEEE

[23] K. Chen et al, ”Efficient SIMD optimization of HEVC encoder over X86 processors”,

