Fast Exclusion of Individual Angular Intra Prediction Modes in HEVC Using Reference Sample Variance

Forschungspraktikum

Chair of Multimedia Communications and Signal Processing
Friedrich-Alexander University of Erlangen-Nürnberg

Christoph Pylinski

January 29, 2016

1 Introduction

The next-generation video coding standard High Efficiency Video Coding (HEVC) aims to tackle two main goals. These include significant improvements in coding efficiency in contrast to H.264/AVC as well as low enough complexity in order to operate in various use cases, such as mobile environments [1]. As described in detail in [1], intra coding in HEVC is based on spatial sample prediction, which is followed by transform coding. The goal of this internship was to investigate further if the 33 angular intra prediction modes used in HEVC can be rapidly excluded individually based on a principle that is similar to [2]. The intra coding design in HEVC provides fundamental elements, such as planar prediction, to generate smooth sample surfaces, angular prediction with 33 prediction directions, and more [1]. However, this report focuses solely on the 33 angular prediction modes and how to determine if each one shall be excluded or not. An overview of the arrangement of these angular intra prediction modes is provided by Figure 2 in [1]. The 33 angular intra prediction modes are designated to optimize the prediction accuracy even when natural illustrations often contain vertical and horizontal patterns [1]. In general, angular intra prediction modes are utilized to continue structure within the reference samples into the prediction block and if the image does not consist of directional structure, these angular prediction modes may be neglected prior to more computational intensive encoding steps [3]. In order to understand the reference sample windows and their indices it is referred to Figure 1 in [2], where a block of size N $\times$ N is illustrated with the corresponding 4N + 1 reference samples.

2 Individual Angular Mode Exclusion

The proposed algorithm excludes individual angular prediction modes whenever the corresponding reference sample windows are without any structure and therefore with a low variance value. Each angular mode can be referred to a certain area of reference samples. The starting point i_{min} and the ending point i_{max} of the selected reference sample area spans a window. These window indices are made available in look-up tables. As mentioned previously, the angular intra prediction modes are used to continue structures within the reference

samples into the prediction block. Hence, a structured prediction is unfeasible whenever these reference samples do not accommodate structure [2]. The following subsections describe the window size calculation and the proposed algorithm along with a descriptive algorithm scheme in order to understand how the algorithm operates.

2.1 Angular Mode Clustering and Calculation of Window Sizes

Firstly, the 33 angular intra prediction modes are clustered in a horizontal prediction group and a vertical prediction group in which each mode corresponds to a projection displacement d as depicted in Table 1 and 2. The intra prediction mode numbering starts at 0 for Planar and 1 for DC and increments further for the residual 33 modes [1]. For visualization of the direction of each angular intra prediction mode [1] can be consulted.

Horizontal Prediction																
Mode	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
d	+32	+26	+21	+17	+13	+9	+5	+2	0	-2	-5	-9	-13	-17	-21	-26

Table 1: HEVC angular intra prediction modes from 2 to 17 to indicate horizontal directionalities.

Vertical Prediction																	
Mode	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
d	-32	-26	-21	-17	-13	-9	-5	-2	0	+2	+5	+9	+13	+17	+21	+26	+32

Table 2: HEVC angular intra prediction modes from 18 to 34 to indicate vertical directionalities.

With this background knowledge, the look-up tables can be determined utilizing (1), (2), (3), (4), where N is the current block size and d is the projection displacement. As a result, these equations provide the indices i_{min} and i_{max} for each window in each group and solely include a certain scope of the reference sample stripe.

$$i_{min,hor}(d,N) = \begin{cases} 1 + \left\lfloor \frac{d}{32} \right\rfloor & \text{for } d \in [0;+32] \\ 1 + \left\lfloor \frac{Nd}{32} \right\rfloor & \text{for } d \in [-32;-2] \end{cases} \tag{1}$$

$$i_{max,hor}(d,N) = \begin{cases} N + \left\lceil \frac{Nd}{32} \right\rceil & \text{for } d \in [0;+32] \\ N + \left\lceil \frac{d}{32} \right\rceil & \text{for } d \in [-32;-2] \end{cases} \tag{2}$$

$$i_{min,ver}(d,N) = \begin{cases} 1 + \left\lfloor \frac{d}{32} \right\rfloor & \text{for } d \in [0;+32] \\ 1 + \left\lfloor \frac{Nd}{32} \right\rfloor & \text{for } d \in [-32;-2] \end{cases} \tag{3}$$

$$i_{max,ver}(d,N) = \begin{cases} N + \left\lceil \frac{Nd}{32} \right\rceil & \text{for } d \in [0;+32] \\ N + \left\lceil \frac{d}{32} \right\rceil & \text{for } d \in [-32;-2] \end{cases} \tag{4}$$

For example, Table 3 represents the indices and window widths for all modes in the horizontal group for a block size N equal to 4. The employed look-up tables, however, contain this information for $N = 4, 16, 32, 64$ and can be consulted from the Matlab script *mode2window_mapping.m* which is appended to this report.

Window indices i_{min} and i_{max} for horizontal prediction																
Mode	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
i_{min}	2	1	1	1	1	1	1	1	1	0	0	-1	-1	-2	-2	-3
i_{max}	8	8	7	7	6	6	5	5	4	4	4	4	4	4	4	4
b	7	8	7	7	6	6	5	5	4	5	5	6	6	7	7	8

Table 3: Window indices i_{min} and i_{max} for block size $N = 4$ and resulting window width b.

2.2 Proposed Algorithm

Based on the knowledge of the calculated window widths, the involved reference samples may be analyzed in order to compute their variance σ^2. Due to the fact that a structured prediction is unfeasible whenever the corresponding reference samples do not accommodate structure, it is necessary to verify this by evaluating the variance of the current window [2]. If this examination of the current window detects structure, then this particular window has to be evaluated further. In fact, if the variance of all reference samples wrapped by the current window exceeds a threshold T, the corresponding angular intra prediction mode will not be excluded for further processes. The following equations depict the computation of the variance of each window in its group where $R_{i,0}$ and $R_{0,j}$ are the corresponding horizontal and vertical reference samples, respectively.

$$\sigma^2_{window,hor} = \left(\frac{1}{b} \cdot \sum_{i=i_{min}}^{i_{max}} R^2_{i,0} \right) - \mu^2_R \tag{5}$$

$$\sigma^2_{window,ver} = \left(\frac{1}{b} \cdot \sum_{j=j_{min}}^{j_{max}} R^2_{0,j} \right) - \mu^2_R \tag{6}$$

The algorithm below summarizes the implementation of the entire algorithm for individual angular mode exclusion. The function *checkIndividualAngles(...)* is implemented in the HEVC reference software HM-16.2. Additionally, this function includes another function *getVariance(...)* which returns the variance of each angular mode. As described previously, if the returned variance is greater than the provided threshold T the corresponding angular mode is set "true" and is kept for further use. Prior to the consideration of the two threshold extreme cases, it should be noted that, by adjusting the threshold T, it is feasible to accelerate the encoder adaptation as well as the allowed bitrate increase [2]. In the first extreme case the threshold is set to zero and leads to no exclusion of any angular prediction mode. The second extreme case sets the threshold high enough to permanently exclude angular prediction modes and should do so in the fastest possible encoding time.

```
Result: List of all angular modes which are either true or false, i.e. if an angular mode
        is set true it will not be excluded.
for each modeIndex do
    if (modeIndex < 18) then
        // Horizontal Group
        call getVariance(modeIndex)
            return σ²[modeIndex];
        if σ²[modeIndex] < T then
            // exclude corresponding angular mode
            angularMode[modeIndex] = false;
        else
            // keep corresponding angular mode
            angularMode[modeIndex] = true;
        end
    else
        // Vertical Group
        call getVariance(modeIndex)
            return σ²[modeIndex];
        if σ²[modeIndex] < T then
            // exclude corresponding angular mode
            angularMode[modeIndex] = false;
        else
            // keep corresponding angular mode
            angularMode[modeIndex] = true;
        end
    end
end
```

Algorithm 1: Illustration of the implemented algorithm to exclude individually angular intra prediction modes. This algorithm represents the function *checkIndividualAngles(...)*.

3 Evaluation and Results

This evaluation uses a quantization paramater (QP) set of $QP = 22, 27, 32$, and 37 as in [4] in order to obtain the bitrate and quality values. For the simulations, 10 frames of the HEVC test sequences form *ClassB*, *ClassC*, *ClassD*, and *ClassE* are encoded and the *intra mode* configuration is selected [4].

This evaluation is compared to the unmodified HM-16.2 Encoder. Illustrations of the encoder performance regarding Δ bitrate and Δ encoding time are depicted in Figure 1 in which the threshold ranges from 0 to 150 and where the step-size is set to 10 for values up to 100. Each subfigure demonstrates the average bitrate changes as well as the differences of the encoding time for the previously listed sequence classes. The diagrams for all sequence classes prove that there is no change to the original HM-16.2 encoding algorithm for $T = 0$. A smaller step-size of the threshold in the range 0 to 100 is chosen to observe the roughly linear increase in bitrate in this area as well as to identify the high time reduction up to an approximate threshold value of $T = 50$. In addition, above $T = 50$ it can be noticed that the encoding time curve becomes less steep. This observation leads to a reasonable trade-off between additional bitrate and time reduction when selecting a threshold in the area of $T = 50$.

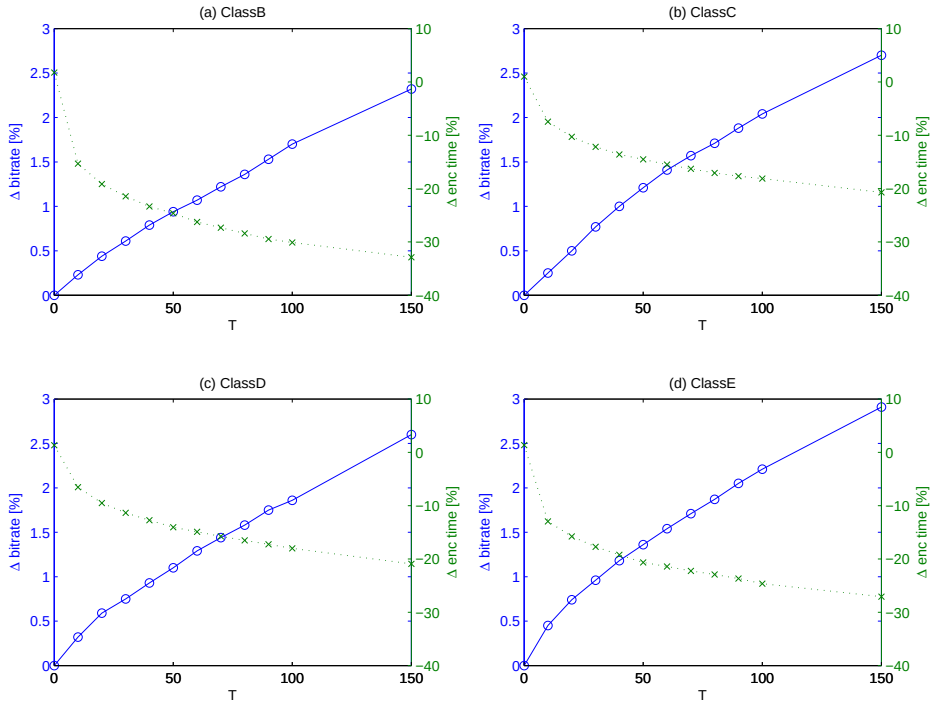

Figure 1: Encoder performance showing Δ bitrate and Δ encoding time over threshold T.

		$T = 30$		$T = 50$		$T = 70$		$T \to \infty$	
		Δ bitrate	Δ enc time	Δ bitrate	Δ enc time	Δ bitrate	Δ enc time	Δ bitrate	Δ enc time
ClassB	*Kimono1*	0.27	-26.25	0.44	-30.64	0.65	-34.44	3.63	-50.18
	ParkScene	0.37	-20.39	0.50	-24.07	0.63	-26.57	3.46	-48.87
	Cactus	0.59	-18.56	0.88	-20.74	1.25	-23.51	14.70	-48.49
	BasketballDrive	1.40	-30.55	2.21	-33.35	2.70	-35.47	20.71	-49.22
	BQTerrace	0.42	-11.54	0.68	-14.34	0.89	-16.69	15.97	-47.95
	average	0.61	-21.46	0.94	-24.63	1.22	-27.34	11.69	-48.94
ClassC	*BasketballDrill*	1.46	-18.96	2.42	-22.62	3.21	-24.71	31.70	-47.60
	BQMall	0.86	-11.89	1.26	-13.76	1.53	-15.36	20.68	-47.23
	PartyScene	0.32	-4.48	0.49	-6.07	0.64	-7.27	11.65	-47.01
	RaceHorses	0.43	-13.33	0.69	-15.59	0.92	-17.80	11.26	-47.85
	average	0.77	-12.17	1.21	-14.51	1.57	-16.29	18.82	-47.42
ClassD	*BasketballPass*	1.16	-16.85	1.63	-19.97	2.27	-22.00	20.74	-47.36
	BQSquare	0.95	-11.30	1.24	-12.88	1.43	-13.44	14.71	-48.22
	BlowingBubbles	0.45	-7.84	0.71	-10.89	1.01	-13.11	12.42	-47.49
	RaceHorses	0.44	-9.44	0.83	-12.46	1.07	-14.50	13.90	-46.90
	average	0.75	-11.36	1.10	-14.05	1.44	-15.76	15.44	-47.49
ClassE	*FourPeople*	1.31	-20.21	1.73	-23.60	2.16	-23.81	21.31	-48.10
	Johnny	1.97	-29.30	2.53	-33.06	2.90	-33.06	28.47	-48.67
	KristenAndSara	1.90	-28.15	2.30	-31.78	2.73	-31.99	26.98	-48.66
	average	1.73	-25.89	2.19	-29.48	2.60	-29.62	25.59	-48.48
	total average	0.96	-17.72	1.36	-20.67	1.71	-22.25	17.89	-48.08

Table 4: Detailed results of the Δ bitrate and the Δ encoding time (both are given in %) for three exemplary threshold values $T = 30$, $T = 50$, and $T = 70$, as well as for $T \to \infty$

Table 4 lists the coding results in detail for all considered sequence classes in a threshold range from 30 to 70 with a step-size of 20 as well as for $T \to \infty$. From this table one can derive that for sequences with higher resolution, such as in *ClassB* and *ClassE*, the average time savings are higher at approximately the same bitrate increase than for the other two classes.

For $T \to \infty$ the average time savings are approximately 48% with an average bitrate increase between 11% and 25%.

4 Conclusion

This report presents an extension of [2] in order to increase the agility of the intra mode decision process in HEVC. If ClassE at a threshold value $T = 50$ is considered in greater detail and compared to the results from [2], the encoding time may be reduced by approximately 0.2%, however, the bitrate loss increases to about 1.14%. Due to this fact, the selected reference samples have to be evaluated in a different way in order to decrease the encoder time at a reasonable bitrate. Without any adaptations, this approach does not significantly raise the performance of the presented approach from [2]. Therefore, [3] presents a solution where the mean of the reference sample windows is used in lieu of the variance. The value of the computed mean is evaluated and decides wether a certain angular mode will achieve a satisfying prediction of the current block or not. Additionally, the reference samples are weighted in order to consider the different number of predicted samples of some reference samples [3]. Further details can be consulted from [3].

References

[1] J. Lainema, F. Bossen, W.-J. Han, J. Min, and K. Ugur, "Intra coding of the HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, pp. 1792–1801, Dec. 2012.

[2] A. Heindel and A. Kaup, "Fast exclusion of angular intra prediction modes in HEVC using reference sample variance," Montreal, Canada, May 2016, to appear in Proc. IEEE International Symposium on Circuits and Systems (ISCAS).

[3] A. Heindel, C. Pylinski, and A. Kaup, "Two-stage exclusion of angular intra prediction modes for fast mode decision in HEVC," Phoenix, Arizona, USA, 2016, submitted to IEEE International Conference on Image Processing (ICIP).

[4] F. Bossen, "Common test conditions and software reference configurations," document JCTVC-L1100, ITU-T VCEG and ISO/IEC MPEG (JCT-VC), Geneva, Switzerland, Jan. 2013.

5 Appendix

5.1 Mode2Window Matlab Script

```
    % Please enter a block size s to either 4, 8, 16, 32, or 64
s = 4;

%% Horizontal Prediction
angular_mode_hor = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17];
d_hor = [32, 26, 21, 17, 13, 9, 5, 2, 0, -2, -5, -9, -13, -17, -21, -26];

i_min_hor = [];
i_max_hor = [];

for i=1:length(d_hor)
    j = d_hor(i);  % current entry of list d
    if (j < 0)
        % Calculate i_max values and add them to the i_max_hor list
        max = s + ceil(j/32);
        i_max_hor(end+1) = max;
        % Calculate i_min values and add them to the i_min_hor list
        min = 1 + floor((s*j)/32);
        i_min_hor(end+1) = min;
    else
        % Calculate i_max values and add them to the i_max_hor list
        max = s + ceil((s*j)/32);
        i_max_hor(end+1) = max;
        % Calculate i_min values and add them to the i_min_hor list
        min = 1 + floor(j/32);
        i_min_hor(end+1) = min;
    end
end
% Calculate each window width
hor_window_width = [];
for i=1:length(i_min_hor)
    min = i_min_hor(i);
    max = i_max_hor(i);
    if (min>=0)
        width = max-min +1;
        hor_window_width(end+1) = width;
    else
        width = max+abs(min) +1;
        hor_window_width(end+1) = width;
    end
end
% All information for current block size
% 1st row shows angular mode index
% 2nd row shows prediction displacement d
% 3rd row shows i_min
% 4th row shows i_max
% 5th row shows width of each window
all_data_hor = [angular_mode_hor ; d_hor ; i_min_hor ; i_max_hor ; hor_window_width];

% output in all_data_hor.txt
fileID = fopen('all_i_hor.txt','w');
fprintf(fileID, '%i, ', i_min_hor);
fprintf(fileID, '\n');
fprintf(fileID, '%i, ', i_max_hor);
fclose(fileID);
%% Vertical Prediction
```

```
angular_mode_ver = [18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34];
d_ver = [-32, -26, -21, -17, -13, -9, -5, -2, 0, 2, 5, 9, 13, 17, 21, 26, 32];

i_min_ver = [];
i_max_ver = [];

for i=1:length(d_ver)
    j = d_ver(i);
    if (j < 0)
        % Calculate i_max values and add them to the i_max_ver list
        max = s + ceil(j/32);
        i_max_ver(end+1) = max;
        % Calculate i_min values and add them to the i_min_ver list
        min = 1 + floor((s*j)/32);
        i_min_ver(end+1) = min;
    else
        % Calculate i_max values and add them to the i_max_ver list
        max = s + ceil((s*j)/32);
        i_max_ver(end+1) = max;
        % Calculate i_min values and add them to the i_min_ver list
        min = 1 + floor(j/32);
        i_min_ver(end+1) = min;
    end
end
% Calculate each window width
ver_window_width = [];
for i=1:length(i_min_ver)
    min = i_min_ver(i);
    max = i_max_ver(i);
    if (min>=0)
        width = max-min +1;
        ver_window_width(end+1) = width;
    else
        width = max+abs(min) +1;
        ver_window_width(end+1) = width;
    end
end
% All information for current block size
% 1st row shows angular mode index
% 2nd row shows prediction displacement d
% 3rd row shows i_min
% 4th row shows i_max
% 5th row shows width of each window
all_data_ver = [angular_mode_ver ; d_ver ; i_min_ver ; i_max_ver ; ver_window_width];

% output in all_data_hor.txt
fileID = fopen('all_i_ver.txt','w');
fprintf(fileID, '%i, ', i_min_ver);
fprintf(fileID, '\n');
fprintf(fileID, '%i, ', i_max_ver);
fclose(fileID);
```