Chapter 5

RESEARCH DETAILS

5.1 Introduction

This chapter provides the details of proposed research work to achieve research objectives outlined in third chapter. Aim of the research was to achieve consistent PSNR, reduce in bit rate, reduce in encoding time and achieve higher compression for Intra frame of various yuv test video sequences in QCIF and CIF resolution and in 4:2:0 format at different quantization parameters, various Gaussian values with all intra prediction modes in Advanced Video Coding Standard using new approach i.e. Gaussian pulse. The simulation results of proposed method for Intra frame of different test yuv sequences using new approach are presented and discussed in forthcoming sections.
5.2 Proposed system diagram

The performance parameters such as video quality, bit rate, encode time and compression of proposed research measured not only a result of single coding tool but it requires a more number of coding tools. Figure 5.1 shows the proposed research block diagram consists major functional blocks are Intra prediction Integer transform, Gaussian pulse, quantization, context adaptive variable length coding, inverse quantization, inverse transformation block.

In proposed block diagram there are two data paths such as forward data path and reconstruction data path. In the forward data path, an input intra frame is presented for encoding; each intra frame is processed in terms of a macro block of size 16x16 pixels. Each macro block is further sub divided into 4x4 sub block. Each 4x4 sub block is encoded in intra prediction. A prediction block \( P \) is formed based on a reconstructed block. In intra mode, prediction \( P \) is formed from samples in the current block is based on previously reconstructed block. The prediction \( P \) is subtracted from the current block to produce a residual or difference macro block. This residual is transformed using integer transform; transformed
coefficients are multiplied with generated Gaussian pulse which gives Gaussian coefficients. These coefficients are quantized using quantization which gives quantized transform coefficients. These coefficients are zig zag scanned and reordered. Quantization coefficients are entropy encoded using Context Adaptive Variable Length Coding to get the compressed bit stream. In the reconstruction data path the quantized coefficients are inverse quantized and inverse transformed to produce a residual block. The prediction block $P$ is added to residual block to create a reconstructed block, which further it can use for prediction of other blocks of intra frame. This process repeats till last sub block of intra frame.

5.3 Intra frame coding

In a test yuv video sequence the first frame is an Intra-frame (I-frame). Intra frame contain spatial redundancy. In order to remove spatial redundancy in intra frame, intra prediction process of AVC is used. Prediction is formed within the intra frame using intra prediction modes is known as Intra frame coding. For Intra frame, prediction is performed within the current frame and without using any other frames. Prediction is performed on 4x4 luma blocks of macro block of intra frame.

5.4 Intra Prediction

Intra prediction means predict the similarity between the neighbouring pixels in current frame to improve coding efficiency. It is the first process of Advanced Video Coding Standard which reduces spatial redundancies within the frame. In the proposed work, macro block of intra frame is encoded using 4x4 intra prediction modes. Prediction block of intra frame is obtained by intra prediction modes.
5.4.1 Intra prediction modes

The spatial redundancy in Intra frame can remove by using intra prediction modes. The nine prediction modes are Mode 0, 1, 2, 3, 4, 5, 6, 7 and 8. All nine modes are used for encoding intra 4x4 block of Intra frame. In the proposed work, the prediction block of intra frame obtained by using prediction equations of all 4x4 intra prediction modes. The following subsequent section gives prediction equations of all prediction Modes.

**Vertical Mode (Mode-0)**

In mode 0, reconstructed samples A B C D of previously reconstructed block of intra frame serves as prediction for current block samples of intra frame. In the reconstructed block A to M are reconstructed samples. M and A to H is previously reconstructed pixels of upper macro block, I to L previously reconstructed samples of left macro block and a to p are prediction samples of current block. The pixel positions in a 4x4 block are represents (0, 0) to (3, 3).

The following generated equations of vertical mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
\text{Pred (0, 0)} = (1, 0) = (2, 0) = (3, 0) = A \\
\text{Pred (0, 1)} = (1, 1) = (2, 1) = (3, 1) = B \\
\text{Pred (0, 2)} = (1, 2) = (2, 2) = (3, 2) = C \\
\text{Pred (0, 3)} = (1, 3) = (2, 3) = (3, 3) = D
\]

**Horizontal Mode (Mode-1)**

In mode 1, reconstructed samples I J K L of previously reconstructed block of intra frame serves as prediction for current block samples of intra frame. The following generated equations of horizontal mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
\text{Pred (0, 0)} = (0, 1) = (0, 2) = (0, 3) = I \\
\text{Pred (1, 0)} = (1, 1) = (1, 2) = (1, 3) = J \\
\text{Pred (2, 0)} = (2, 1) = (2, 2) = (2, 3) = K \\
\text{Pred (3, 0)} = (3, 1) = (3, 2) = (3, 3) = L
\]
DC Mode (Mode-2)

DC mode prediction for 4x4 block is to replace all pixels in the current 4x4 block by the mean value of the reconstructed pixels \(A \ B \ C \ D \ I \ J \ K \ L\). The following generated equations of DC mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
\frac{(A+B+C+D+I+J+K+L)}{8}
\]

Diagonal down left mode (Mode-3)

In mode 3, for 4x4 block prediction samples formed from a average weighted of the prediction pixels \(A\) to \(M\). The following generated equations of diagonal down left mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
\begin{align*}
\text{Pred} (0, 0) &= A + 2B + C + 2 >> 2, \\
\text{Pred} (0, 1) &= (1, 0) = B + 2C + D + 2 >> 2 \\
\text{Pred} (0, 2) &= (2, 0) = (1, 1) = C + 2D + E + 2 >> 2 \\
\text{Pred} (0, 3) &= (2, 1) = (1, 2) = (3, 0) = D + 2E + F + 2 >> 2 \\
\text{Pred} (1, 3) &= (2, 2) = (3, 1) = E + 2F + G + 2 >> 2 \\
\text{Pred} (2, 3) &= (3, 2) = F + 2G + H + 2 >> 2 \\
\text{Pred} (3, 3) &= G + 3H + 2 >> 2
\end{align*}
\]

Diagonal down Right mode (Mode -4)

In mode 4, for 4x4 block prediction samples formed from a average weighted of the prediction samples \(A\) to \(M\). The following generated equations of diagonal downright mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
\begin{align*}
\text{Pred} (0, 0) &= (1, 1) = (2, 2) = (3, 3) = A + 2M + I + 2 >> 2 \\
\text{Pred} (0, 2) &= (1, 3) = A + 2B + C + 2 >> 2 \\
\text{Pred} (0, 3) &= B + 2C + D + 2 >> 2 \\
\text{Pred} (3, 0) &= J + 2K + L + 2 >> 2
\end{align*}
\]
Pred (0, 1) = (1, 2) = (2, 3) = M + 2A + B + 2 >> 2
Pred (1, 0) = (2, 1) = (3, 2) = M + 2I + J + 2 >> 2
Pred (2, 0) = (3, 1) = I + 2J + K + 2 >> 2

**Vertical right mode (Mode -5)**

In mode 5, for 4x4 block prediction samples formed from a average weighted of the prediction samples A to M. The following generated equations of diagonal vertical mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

Pred (0, 0) = (2, 1) = M + A + 1 >> 1
Pred (0, 1) = (2, 2) = A + B + 1 >> 1
Pred (0, 2) = (2, 3) = B + C + 1 >> 1
Pred (0, 3) = C + D + 1 >> 1
Pred (1, 0) = (3, 1) = I + 2M + A + 2 >> 2
Pred (1, 1) = (3, 2) = M + 2A + B + 2 >> 2
Pred (1, 2) = (3, 3) = A + 2B + C + 2 >> 2
Pred (3, 0) = I + 2J + K + 2 >> 2
Pred (1, 3) = B + 2C + D + 2 >> 2
Pred (2, 0) = M + 2I + J + 2 >> 2

**Horizontal down mode (Mode -6)**

In mode 6, for 4x4 luma block prediction samples formed from a average weighted of the prediction samples A to M. The following generated equations of horizontal down mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

Pred (0, 0) = (1, 2) = M + I + 1 >> 1
Pred (0, 1) = (1, 3) = I + 2M + A + 2 >> 2
Pred (0, 2) = B + 2A + M + 2 >> 2
Pred (0, 3) = C + 2B + A + 2 >> 2
Pred (1, 0) = (2, 2) = I + J + 1 >> 1
Pred (1, 1) = (2, 3) = M + 2I + J + 2 >> 2
Vertical left mode (Mode -7)

Mode 7 is the vertical left mode, in mode 7, for 4x4 block prediction samples formed from a average weighted of the prediction pixels $A$ to $M$. The following generated equations of horizontal down mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
P_{red}(0, 0) = A + B + 1 >> 1
\]
\[
P_{red}(0, 1) = (2, 0) = B + C + 1 >> 1
\]
\[
P_{red}(0, 2) = (2, 1) = C + D + 1 >> 1
\]
\[
P_{red}(0, 3) = (2, 2) = D + E + 1 >> 1
\]
\[
P_{red}(1, 0) = A + 2B + C + 2 >> 2
\]
\[
P_{red}(1, 1) = \text{pred}(3, 0) = B + 2C + D + 2 >> 2
\]
\[
P_{red}(1, 2) = (3, 1) = C + 2D + E + 2 >> 2
\]
\[
P_{red}(1, 3) = (3, 2) = D + 2E + F + 2 >> 2
\]
\[
P_{red}(2, 1) = E + F + 1 >> 1
\]
\[
P_{red}(3, 3) = E + 2F + G + 2 >> 2
\]

Horizontal up mode (Mode -8)

In mode 8, for 4x4 luma block prediction samples formed from a average weighted of the prediction pixels $A$ to $M$. The following generated equations of horizontal down mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[
P_{red}(0, 0) = I + J + 1 >> 1
\]
\[
P_{red}(0, 1) = I + 2J + K + 2 >> 2
\]
\[
P_{red}(0, 2) = (0, 2) = (1, 0) = J + K + 1 >> 1
\]
\[
P_{red}(0, 3) = (1, 1) = J + 2K + L + 2 >> 2
\]
\[
P_{red}(1, 2) = (2, 0) = K + L + 1 >> 1
\]
Pred (1, 3) = (2, 1) = K + 3L + 2 >> 2
Pred (2, 2) = (2, 1) = (2, 2) = (2, 3) = (3, 0) = (3, 1) = (3, 2) = (3, 3) = L

Obtained the prediction blocks using prediction equations of all intra prediction modes, these prediction blocks are subtracted with current blocks of intra frame.

**Residual Block**

Residual block is obtained by subtracting prediction block with current block of intra frame.

**Integer transform**

The residual data which is given to transform block. The transform block will convert the residual data into transformed coefficients in the form of frequency components. Apply 4x4 integer transform for the residual data [104]. Integer transform equation for the residual data is given by.

\[
W = AXA^T \quad \text{Equation 5.1}
\]

Above equation expressed in matrix form

\[
W = A \cdot X \cdot A^T = \begin{bmatrix}
1 & 1 & 1 & 1 \\
2 & 1 & -1 & -2 \\
1 & -1 & -1 & 1 \\
1 & -2 & 2 & -1 \\
\end{bmatrix} \cdot \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & -1 & -2 \\
1 & -1 & -1 & 2 \\
1 & -2 & 1 & -1 \\
\end{bmatrix}
\]

Where \(X\) is the input 4x4 residual data, \(W\) is the matrix of transformed coefficients, \(A\) integer matrix and \(A^T\) transpose of \(A\) are integer matrix or constant modified DCT matrices i.e. contain integers.

**Operation**

Multiply the first row of integer matrix \((A)\) with each column of the residual values \((X)\) then it generate first row of products which is multiplied with each column of the transpose of integer matrix \((A^T)\) to produce the first row of integer transformed coefficients. Repeat the procedure for remaining rows of integer matrix with each column of residual and transpose
matrix to get remaining transform coefficients. For residual block, obtained transformed coefficients using integer transform. These transformed coefficients are multiplied with generated Gaussian pulse $G(v)$ at macro block level.

In proposed work the consistent PSNR, reduce in bit rate and more compression obtained by using new approach Gaussian pulse.

The following section gives details about generation Gaussian pulse and its operation with transform coefficients of intra frame. This Gaussian pulse plays an important role for achievement research objectives.

## 5.5 Gaussian pulse

A pulse has the waveform of a Gaussian distribution, which is similar to a bell curve shape. Gaussian pulse is shaped as a Gaussian function or Gaussian distribution and is obtained by a Gaussian filter. The Gaussian filter is a windowed filter used to blur images and remove noise and detail. It used to smooth the images it also provides better suppression of higher frequencies than the other filters. Gaussian filter can be either one or two dimensional. The two dimensional Gaussian filters are used to produce Gaussian blurs in image and video processing. The Gaussian function is defined in one-dimension, two-dimension and N-dimension respectively.

One dimensional Gaussian function is

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma^2}.e^{-\left(\frac{x^2}{2\sigma^2}\right)} \quad \text{Equation 5.2}$$

The term $1/\sqrt{2\pi}\sigma$ of one dimensional Gaussian function is the normalization constant, where $x$ is distance in horizontal axis from the origin, $\sigma$ is the standard deviation. The $\sigma$ determines the width of the Gaussian, in statistics $\sigma$ is called the standard deviation and the square of it, $\sigma^2$ is the variance. The scale can only take positive values, $\sigma > 0$, $\sigma$ can never become zero. With the normalization constant the Gaussian function is a normalized function, which is its integral over its full domain is unity for every $\sigma$. increasing the $\sigma$ of the Gaussian function reduces the amplitude substantially [86-90]. The area under curve is unity when Gaussian function is
normalized. So that normalization ensure that grey level remain of picture same when blur the picture with this Gaussian function. Figure 5.2 shows one dimensional Gaussian distribution.

![Figure 5.2 One dimensional Gaussian distribution](image)

The equation of two dimensional Gaussian functions is product of two one dimension of Gaussian function required for image, which is given by.

\[
G(x, y) = \frac{1}{2\pi\sigma^2}.e^{-\left(x^2 + y^2 / 2\sigma^2\right)} \quad \text{Equation 5.3}
\]

Where is \(x\) the distance in the horizontal axis from the origin, \(y\) is the distance in the vertical axis from the origin. Figure 5.3 shows a graphical representation of the two dimensional Gaussian distribution.
The N dimensional Gaussian function is given by

\[ G(x, \sigma) = \frac{1}{(\sqrt{2\pi\sigma})^N} e^{-\frac{|x|^2}{2\sigma^2}} \]  \textbf{Equation 5.4}

The Fourier transform of a Gaussian function for frequencies which remove undesired high frequency components. The cut-off frequency of filter depends on the scale of the Gaussian function. The Gaussian function is the only function for which the Fourier transform has the same shape. One of the important properties of the Gaussian function is that both it and its Fourier transform are real valued. Gaussian is the impulse of a Gaussian filter is more effective at smoothing images [91-95]. In various research areas the Gaussian function is used.

- It used to blur images and remove noise and detail
- It is used to describe normal distributions
- It serves as Gaussian filter in signal processing
- It used in image processing for image smoothen
- It is used to solve diffusion equations in mathematics
- For noise or data it describe a probability distribution
- It is used to define some types of artificial neural networks
- Derivates of Gaussian is used for defining types of visual operations
- Gaussian functions are used in microwave, optical and digital communication systems
• It is a shaping function because it gives a particularly compact frequency-domain excitation

The Gaussian pulse has the following important properties in image or video processing applications.

• Gaussian pulse removes high-frequency components from the image/frames
• Larger Gaussian scaling parameter remove more details it get distorted
• Combination of two pulses or multiply two Gaussian pulses is called Gaussian filter
• It is separable
• It is a symmetric function
• It has frequency-selective excitation
• The Gaussian function is the Fourier of the Gaussian is itself a Gaussian

5.5.1 Generation of Gaussian pulse

Gaussian pulse is based on Gaussian function; it is bell shape curve which is symmetrical that quickly falls off towards zero.

In the proposed research the Gaussian pulse equation used for Intra frame coding in advanced video coding standard to achieve research objective.

\[ G(v) = \exp(-v^2) \quad \text{Equation 5.5} \]

The product of two Gaussian pulses is also Gaussian which is given by.

\[ G(v)' = G(v)*G(v) \quad \text{Equation 5.6} \]

The above equations are used for intra frame coding. After transform, frequency domain samples are obtained, these samples are multiplied with a Gaussian pulse. The Gaussian operation smoothen the samples.
In the proposed research the Gaussian pulses generated for different Gaussian parameters are shown in Figure 5.4 to 5.7. MATLAB code written for generation of Gaussian pulses for different Gaussian values, the generation of Gaussian pulse is considering the sampling frequency, time base and Gaussian parameter. The generated Gaussian pulses are used for multiplication of each transformed coefficients at macro block levels, each such multiplication scales the information content of the signal in a reversible way. The resulting signal would turn abstract.

Figure 5.4 Gaussian pulses for scaling parameter \( v = 0.1, 0.2 \) and 0.4

Figure 5.5 Gaussian pulses for scaling parameter \( v = 0.5, 0.6 \) and 0.8
Figure 5.6 Gaussian pulses for scaling parameter $v = 1, 2$ and $3$

Figure 5.7 Gaussian pulses for scaling parameter $v = 4, 5$ and $6$

The graph of a Gaussian bell shape curve, the Gaussian smooth to an image is convolving the image with a Gaussian function values, i.e. Gaussian smooth reducing the image high-frequency components.

5.5.2 Importance of Gaussian pulse

In order to avoid drastic reduction of PSNR for a coarse quantization step in video coding standard, Gaussian pulse is taken. This pulse made the frequency domain samples are abstract
in a known and controllable manner without intermixing of information. It results reconstructed image would have higher PSNR; bit rate i.e. avoids drastic reduction in the PSNR of the reproduced image and achieves high compression. The Figure 5.8 shows an example of Gaussian smooth with Gaussian pulse.

![Figure 5.8 Gaussian smooth](image)

Gaussian function gives a frequency- domain pattern; Gaussian is a good choice of shaping function since it provides a particularly compact frequency- domain excitation spectrum. It is useful for the majority of applications of frequency- selective excitation. The derivatives of Gaussian pulse are obtained by the derivation of Gaussian pulse. Taking the derivative of a Gaussian pulse increases the number of peaks and valleys of the curve effectively increasing the pulse train frequency and moving the centre frequency higher. The main characteristics of Gaussian pulse are narrow width in time domain and wider width in frequency domain. Gaussian pulse and its wide spectrum are easy to generate. The Gaussian is real, so its phase is zero. In time domain a Gaussian transform to a Gaussian. In frequency domain the spectral component of Gaussian is zero [97-100].

Gaussian coefficients are obtained by multiplying Gaussian distribution with transformed coefficients at macro block level which are in frequency domain samples. Multiplying the generated Gaussian pulse with integer transformed coefficients of macro block level which is in terms of 4x4 sub block then generate abstract frequency domain samples. It avoids intermix of the frequency domain information among the samples. This multiplication operation performs or scales the information content of the signal in a reversible way and avoids drastic reduction of quality of the picture and also improves functionality of quantizer. The degree of
abstraction can controlled through a simple parameter-that is Gaussian scaling parameter. In proposed work, the purpose of the Gaussian pulse is obtaining consistent PSNR and reduces in bit rate. This Gaussian pulse avoids mixing of the frequency domain information among the samples and it recovers the information content in reversible way. The user demands for constant PSNR and good delivery of video quality so in order to deliver better video quality and to get constant PSNR the optimized solution obtained for getting constant or consistent PSNR.

5.5.3 The consistent PSNR

Quality fluctuation has a major negative effect on perceptive video quality, video quality smoothing works target on consistent distortion (i.e., consistent PSNR) and bit rate control, which is to achieve the best-perceived video quality. The image smoothing can be done by Gaussian function at macro block level after transformation. Gaussian smoothing can achieve consistent picture quality. To get consistent PSNR taking the ratio of coefficients of Integer Transform (IT) to the difference of integer transform (coefficients) and window (Gaussian function) of integer transform. The consistent PSNR can be obtained by following

\[
\text{Consistent PSNR} = \frac{(\text{Integer transform})}{(\text{Integer transform}) - (\text{Integer transform}) \times (\text{Gaussian pulse})}
\]

Window filters redistributes or intermixes the frequency domain information among the samples and make them abstract. Each sample is made to depend up on a large set of other samples. These samples are quantised with different step sizes. This prevents a particular set of sample of the original image getting a bad hit. They can enjoy the low quantisation steps which otherwise would have been impossible. However, the information loss in one or the other quantisation step is still substantial. It is only distributed or shared across the samples.

In the proposed work, Gaussian pulse made the frequency domain samples abstract in a known and controllable manner without intermixing of information. The Gaussian pulse is given by

\[
G (v) = \exp (-v^2) \quad \text{Equation 5.7}
\]
The Gaussian pulse provides a mechanism for generating abstract frequency domain samples. The degree of abstraction may be controlled through a simple Gaussian values. For different Gaussian values the bit rate gets traded with the PSNR or quality of the image. After frequency domain transformation, the samples are multiplied with a Gaussian pulse. This operation smoothen the signal. Each such multiplication scales the information content of the signal in a reversible way. The resulting signal would turn abstract. The product of 2 Gaussian pulses is also Gaussian. This way, the abstraction level can be controlled on Gaussian value. After multiplication of Gaussian pulse, the resulting block is Gaussian coefficients, these coefficients are quantized.

Quantization

Quantization process used to remove less important coefficients by dividing each coefficient by an integer. The use of quantization is achieves high compression and discards unnecessary information which is not visual importance.

Gaussian transformed coefficients are quantized by quantization operation to obtain quantized coefficients. The quantization operation is given as shown

\[ Z = \text{round} \left( \frac{GN}{Q\text{step}} \right) \]  \hspace{1cm} \text{Equation 5.8}

Where Z is quantized coefficients, W is transformed coefficients and Qstep is quantization step.

H.264/AVC standard supports a total of 52 values of quantization step (Qstep). It uses a scalar quantizer, which quantize Gaussian transform coefficients. The quantization step i.e. Qstep which is indexed by a Quantization Parameter (QP). Quantization step doubles in size for every increment of 6 in quantization parameter. An increase in quantization parameter can reduces up to 12.5% of the bit-rate. The table 5.1 shows values of quantization step corresponding to each quantization parameter [104].
<table>
<thead>
<tr>
<th>QP</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qstep</td>
<td>0.625</td>
<td>0.6875</td>
<td>0.8125</td>
<td>1</td>
<td>1.125</td>
<td>1.25</td>
<td>1.375</td>
<td>1.625</td>
<td>1.75</td>
<td>2</td>
<td>2.25</td>
<td>2.5</td>
<td>---</td>
</tr>
</tbody>
</table>

Table 5.1 H.264/AVC Quantization step sizes

The unscaled Gaussian transformed coefficients are quantized and scaled in single operation by the following equation.

\[ Z = \text{round} \left( \frac{\text{GN}}{(PF/Qstep)} \right) \]  \textbf{Equation 5.9}

Where PF is the post scaling parameter, table 5.2 show post scaling parameter

<table>
<thead>
<tr>
<th>Position</th>
<th>Post scaling Parameter (PF)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0),(2,0),(0,2) or (2,2)</td>
<td>(a^2)</td>
</tr>
<tr>
<td>(1,1),(1,3),(3,1) or (3,3)</td>
<td>(b^2/4)</td>
</tr>
<tr>
<td>Other</td>
<td>(ab/2)</td>
</tr>
</tbody>
</table>

Table 5.2 Post Scaling Parameter

In H.264/AVC reference model software, the parameter (PF/Qstep) is implemented as multiplication by multiplication parameter and right shift operation. This avoids division operations. The quantization operation modified as

\[ Z = \text{round} \left( \frac{\text{GN} \cdot (MF/2^{qbits})}{Qstep} \right) \]  \textbf{Equation 5.10}

Where \(qbits = 15 + \text{floor}(QP/6)\), \(MF/2^{qbits} = PF/Qstep\), MF is a Multiplication Parameter which is specified in H.264/AVC standard. The multiplication parameter is shown in table 5.3.

<table>
<thead>
<tr>
<th>QP</th>
<th>Positions</th>
<th>Positions</th>
<th>Other positions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(0,0),(2,0),(2,2),(0,2)</td>
<td>(1,1),(1,3),(3,1),(3,3)</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>13107</td>
<td>5243</td>
<td>8066</td>
</tr>
<tr>
<td>1</td>
<td>11916</td>
<td>4660</td>
<td>7490</td>
</tr>
<tr>
<td>2</td>
<td>10082</td>
<td>4194</td>
<td>6554</td>
</tr>
<tr>
<td>3</td>
<td>9362</td>
<td>3647</td>
<td>5825</td>
</tr>
<tr>
<td>4</td>
<td>8192</td>
<td>3355</td>
<td>5243</td>
</tr>
<tr>
<td>5</td>
<td>7282</td>
<td>2893</td>
<td>4559</td>
</tr>
</tbody>
</table>

Table 5.3 Multiplication parameter of H.264/AVC
Quantization process results quantized coefficients, these quantized coefficients are Zig Zag scanned and reordered. The reordered quantized coefficients are given Context Adaptive Variable Length encoder to get compressed bit stream.

5.6 Context Adaptive Variable Length Coder (CAVLC)

In the research work Context based Adaptive Variable-Length Coding entropy encoder is used. It is based on baseline profile. It is used to encode quantized coefficients to obtain compressed bit stream [103]. It is processed into five major steps.

- The number of non coefficients and the trailing ones are encode
- The sign of each trailing ones are encode
- The level of non-zero coefficient are encoding
- The total number of zeros are encode before the last coefficient
- Each run of zeros are encode

CAVLC is used to encode the 4x4 block quantized transform coefficient and assigns variable length codes. The Figure 5.9 shows that steps involved in CAVLC.

<table>
<thead>
<tr>
<th>No. of Non zero coeffs and trailing one’s</th>
<th>Sign of each trailing 1’s</th>
<th>Level of Non zero Coeffs</th>
<th>Total no. of zero before Last coeffs</th>
<th>Zero runs</th>
</tr>
</thead>
</table>

**Figure 5.9** Steps involved for CAVLC

Encoding operation for 4x4 quantized coefficients block using CAVLC process as follows.

- **Encoding the number of coefficients and trailing ones**
  Total Coeffs (non-zero coefficients) and number of trailing +/- 1 value (T1) are encoding.

- **Encoding of sign of trailing ones**
  The sign of trailing ones is encoding with single bit, + sign encoded with 0 and – sign encoded with 1, trailing ones are encoding in reverse order.
• **Encoding the level**
  In reverse order the level of each non-zero coefficients is encoding with the highest frequency.

• **The total number of zeros are encoding before the last coefficient**
  In the reordered sequence of the sub block, total number zeros are encoding before last coefficient.

• **Encoding each of zeros**
  The number of zeros is encoded in reverse order for each non zero coefficient i.e. run\_before. The run\_before is encoded for each non-zero coefficient, starting with highest frequency.

The following examples are to obtain compressed bit using encoding operation of CAVLC

Consider 4x4 block quantized values

\[
\begin{array}{cccc}
0 & 3 & -1 & 0 \\
0 & -1 & 1 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{array}
\]

The reordered sequences of the 4x4 quantized sub blocks are as follows

\[0, 3, 0, 1,-1,-1, 0, 1, 0 \text{--------}

In the reordered sequence of sub block, total number of non zero coefficients are five which is indexed from highest frequency down to lowest frequency, total number of zeros are three and number of trailing ones (T1) are three, but in the example there are 4 trailing ones but AVC standard specifies only a maximum of three can be encoded. Quantization coefficients are encoded as shown in table 5.4.
Encoding operation

<table>
<thead>
<tr>
<th>Element</th>
<th>Values</th>
<th>code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coeff_token</td>
<td>Total non zero coeffs =5, T1s=3</td>
<td>0000100</td>
</tr>
<tr>
<td>T1 sign (4)</td>
<td>+</td>
<td>0</td>
</tr>
<tr>
<td>T1 sign (3)</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>T1 sign (2)</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>Level (1)</td>
<td>+1</td>
<td>1</td>
</tr>
<tr>
<td>Level(0)</td>
<td>+3</td>
<td>0010</td>
</tr>
<tr>
<td>TotalZeros</td>
<td>3</td>
<td>111</td>
</tr>
<tr>
<td>run_before(4)</td>
<td>run_before=1, Zerosleft=3</td>
<td>10</td>
</tr>
<tr>
<td>run_before(3)</td>
<td>run_before=1, Zerosleft=2</td>
<td>1</td>
</tr>
<tr>
<td>run_before(2)</td>
<td>run_before=0, Zerosleft=2,</td>
<td>1</td>
</tr>
<tr>
<td>run_before(1)</td>
<td>run_before=1, Zerosleft=2,</td>
<td>01</td>
</tr>
<tr>
<td>run_before(1)</td>
<td>run_before=1, Zerosleft=1, For last coefficient code is not required</td>
<td></td>
</tr>
</tbody>
</table>

The encoded bits for 4x4 quantized blocks are 000010001110010111101101.

Consider 4x4 block quantized values

\[
\begin{array}{cccc}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 \\
\end{array}
\]

In the reordered sequence of sub block, the total number of non zero coefficients are three, total number of zeros are seven, number of trailing ones (T1) are three. Encoding of quantization coefficients are shown in table 5.5.
Encoding operation

<table>
<thead>
<tr>
<th>Table 5.5 Encoded quantized coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element</strong></td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>Coeff_token</td>
</tr>
<tr>
<td>T1 sign (2)</td>
</tr>
<tr>
<td>T1 sign (1)</td>
</tr>
<tr>
<td>T1 sign (0)</td>
</tr>
<tr>
<td>TotalZeros</td>
</tr>
<tr>
<td>run_before(2)</td>
</tr>
<tr>
<td>run_before(1)</td>
</tr>
<tr>
<td>run_before(0)</td>
</tr>
</tbody>
</table>

The encoded bits for the 4x4 quantized blocks are **000110001110010**.

Encoded bit stream or compressed bits are obtained using above CAVLC process. The reconstruction of processed sub block is performed as follows.

The quantized coefficients are inverse quantized with the inverse quantization operation

**Inverse Quantization**

The basic inverse quantization operation is

\[ W' = Z \cdot \text{Qstep} \quad \text{Equation 5.11} \]

Where \( Z \) is quantized coefficients, \( \text{Qstep} \) is quantization step.

To avoid rounding errors, pre scaling parameter is included in quantization with addition constant scaling parameter of 64 for inverse transform is given by

\[ W' = Z \cdot \text{Qstep} \cdot \text{PF.64} \quad \text{Equation 5.12} \]
The inverse quantization operation is

\[ W' = Z U \cdot 2^{\left\lfloor \frac{Q}{6} \right\rfloor} \quad \text{Equation 5.13} \]

Where \( W' \) is a scaled coefficient, \( Z \) are quantized coefficients, \( U \) are rescaling parameters \((U = \text{Qstep.PF.64})\). The rescaling parameters are depends upon the position that is specified in the H.264/AVC standard. Rescaling parameters of advanced video coding standard are shown in table 5.6.

<table>
<thead>
<tr>
<th>QP</th>
<th>Positions</th>
<th>Other positions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(0,0),(2,0),(2,2),(0,2)</td>
<td>(1,1),(1,3),(3,1),(3,3)</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>13</td>
<td>20</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>23</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>25</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>29</td>
</tr>
</tbody>
</table>

To obtain rescaling parameters of AVC depends on QP and positions.

Example

For QP=4, then Qstep = 1 and \( 2^{\left\lfloor \frac{4}{6} \right\rfloor} = 2 \). For position \((i, j) = (2, 2)\), then PF = \( a^2 = 0.25 \) then \( U = (\text{Qstep.PF.64}) = 1 \times 0.25 \times 64 \cong 16 \).

The parameter \( 2^{\left\lfloor \frac{Q}{6} \right\rfloor} \) makes the rescaled output increase by a parameter of 2 for every increment of 6 in quantization parameter.

After obtaining the inverse quantization inverse transformed process are performed for inverse quantization.

Inverse transform

The inverse transform operation is

\[ X' = A^\dagger \cdot W' \cdot A \quad \text{Equation 5.14} \]
The above equation is expressed in matrix form

\[
\hat{X} = A_i^\top W' \cdot A_i = \begin{bmatrix}
1 & 1 & 1 & 1/2 \\
1/2 & -1 & -1 & 1 \\
-1/2 & -1 & 1 \\
1 & -1 & 1 & -1/2
\end{bmatrix}
\begin{bmatrix}
1 & 1 & 1 & 1 \\
1/2 & -1/2 & -1 \\
1 & -1 & -1 & 1 \\
1/2 & -1 & 1 & -1/2
\end{bmatrix}
\]

Where \( W' \) inverse quantized coefficients, \( A_i \) is a constant inverse integer matrix, \( A_i^\top \) is a transpose constant inverse integer matrix.

Reconstructed block

Reconstructed block is obtained by adding prediction block to inverse transformed coefficients i.e.

\[
\text{Reconstructed block} = \text{Predicted block} + X'
\]

Where \( X' \) is inverse transformed coefficients

This reconstructed block is serves as reference block for next current block of intra frame this process repeats for all the sub blocks of intra frame.

5.7 Advantages of Intra frame coding

H.264/AVC compression with only intra frame prediction is especially suitable for low cost and low power applications. The following are the advantages of using Intra frame coding.

- Intra frame coding results in less complexity compared to Inter frame coding
- Rate distortion performance is better for low and intermediate resolution sequences
- The benefit of Intra frame provides better compression performance
- Intra frame coding is attractive for error resilience and does not allow error propagation
• Intra frame is required for browsing, random access, and editing of video content since each frame is encoded within frame and without using any other reference frames.
• Intra-frames are needed as starting points for new viewers and also for resynchronization points if the transmitted bits are damaged.
• Intra frame can be used for fast-forward, rewind and random access function.

Intra frame coding can be used for various applications such as digital cinema, satellite, medical imaging as well as video surveillance, low bit rate applications and common intermediate format applications.

5.8 Operation of all intra prediction modes

The 4x4 sub-blocks of macro block of intra frame in Advanced Video Coding are encoded using intra prediction modes with Gaussian function. The following section gives detail of generation of prediction block, residual block, integer transformed (IT), Gaussian, Quantization (Q) to (ITIQ) reconstruction block of Intra frame for individual prediction modes. The prediction block is obtained by intra prediction modes. The prediction block is subtracted with current sub block to get residual block, this residual block is coded followed by integer transform, Gaussian multiplication, quantization, inverse quantized and inverse transform with addition of prediction to get reconstructed block.

5.8.1 Operation involved in Horizontal mode (Mode-1)

The steps performed in the horizontal prediction mode for Intra frame coding as follows:
• Intra frame is processed in terms of macro block by macro block. A macro block has 16x16 pixels, which is further divided into sixteen 4x4 sub macro blocks which is performed from left to right and from top to bottom.
• The 4x4 sub macro blocks named as B0, B1, B2, B3, B4, B5,------------------------B15.
In intra frame the first sub macro block no reconstructed pixels are available to obtain the predicted block, so that predicted block for the first block is considered as a zero pixel values

- First sub macro block processed without using previously reconstructed block
- To predict next sub block, reconstruction of first sub block which serves as prediction block
- In horizontal prediction mode last column pixels of the reconstructed block is used as a prediction for current block

The Figure 5.10 shows working operation of the horizontal prediction mode for intra frame using Gaussian pulse. The working operation of Mode-1 (horizontal Mode) for intra frame coding is carried out by: Intra frame is divided into 16x16 macro blocks, which is further divided into 4x4 sub macro block labelled as B_0,B_1,B_2,B_3,B_4,B_5,B_6,B_7,B_8,B_9,B_10,B_11,B_12,B_13 and B_15. Consider an example block B_5 is a current sub block of intra frame which consists of pixels a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8,a_9,a_{10},a_{11},a_{12},a_{13},a_{14},a_{15} and a_{16}. In the reconstructed pixels of block (B_4) are (A,B,C,D,E,F,G,H,E,F,G,I,J,K,L and M). In reconstructed block (B_4) last column I,J,K,L of reconstructed pixels serves as a prediction for current sub block (B_5). Therefore in horizontal prediction mode the last column pixels I,J,K,L of left sub block of B_4 is used as prediction pixels for current block. The prediction for current sub block is processed using last column of the recently reconstructed sub block (B_4).

Residual pixels are obtained by subtracting the current sub block pixels and predicted pixels. Apply integer transform for residual pixels to obtained transformed coefficients which in the form of frequency samples, these samples are multiplied by generated Gaussian pulse this operation provides an abstract frequency domain samples, the degree of abstraction may be controlled through a simple parameter—the scaling parameter. Multiplication scaling parameter performs the information content of the signal in a reversible way and also improving the functionality of the quantiser. Resulting abstract frequency domain samples are quantized and entropy encoded using context based adaptive variable length coder to get compressed bits. Reconstructed block is obtained through the process of inverse quantization and inverse transform and get quality of picture.
Figure 5.10 Horizontal intra prediction mode for intra frame

Generated equations of diagonal down left mode used for obtaining prediction block which serves as prediction for current block samples of intra frame.

\[ \text{Pred (0, 0)} = (0, 1) = (0, 2) = (0, 3) = I \]
\[ \text{Pred (1, 0)} = (1, 1) = (1, 2) = (1, 3) = J \]
\[ \text{Pred (2, 0)} = (2, 1) = (2, 2) = (2, 3) = K \]
\[ \text{Pred (3, 0)} = (3, 1) = (3, 2) = (3, 3) = L \]

The predicted pixels are: I J K and L and prediction pixels are: a₁ to a₁₅

The residual data obtained by: a₁-I, a₂-I, a₃-I, a₄-I, a₅-J, a₆-J, a₇-J, a₈-J, a₉-K, a₁₀-K, a₁₁-K, a₁₂-K, a₁₃-L, a₁₄-L, a₁₅-L and a₁₆-L. The residual pixels are obtained by subtracting predicted pixels (IJKL) with current sub block pixels then apply the integer transform, Gaussian pulse, quantization and inverse quantization, inverse transform to get reconstructed block.
5.8.2 Operation involved in Vertical mode (Mode-0)

The operation involved in vertical prediction mode for intra frame as shown in Figure 5.11. The reconstructed upper pixels A B C D of reconstructed block acts as prediction pixels for the current sub block. The prediction equations of vertical prediction mode are generated as

\[
\begin{align*}
    a_1 &= a_5 = a_9 = a_{13} = A \\
    a_2 &= a_6 = a_{10} = a_{14} = B \\
    a_3 &= a_7 = a_{11} = a_{15} = C \\
    a_4 &= a_8 = a_{12} = a_{16} = D
\end{align*}
\]

![Diagram of vertical prediction mode for intra frame]

**Figure 5.11 Vertical prediction intra prediction mode for intra frame**

The residual pixels are obtained by: \(a_1\)-A, \(a_2\)-B, \(a_3\)-C, \(a_4\)-D, \(a_5\)-A, \(a_6\)-B, \(a_7\)-C, \(a_8\)-D, \(a_9\)-A, \(a_{10}\)-B, \(a_{11}\)-C, \(a_{12}\)-D, \(a_{13}\)-A, \(a_{14}\)-B, \(a_{15}\)-C and \(a_{16}\)-D. The residual pixels are obtained by subtracting predicted pixels (ABCD) with current sub block pixels then apply the integer transform, Gaussian pulse, quantization and inverse quantization, inverse transform to get reconstructed block.
5.8.3 Operation involved in DC mode (Mode-2)

For DC prediction mode, all the pixels of current sub block is predicted by mean value of upper and left reconstructed pixels of reconstructed block. Prediction equation generated for DC mode as follows.

\[
\frac{(A+B+C+D+I+J+K+L)}{8} = S
\]

Mean value of upper and left reconstructed pixels of reconstructed block is represented by \( S \) which serves as prediction of prediction pixels (\( a_1 \) to \( a_{16} \)). The residual pixels are obtained by:

\[
\begin{align*}
    &a_1 - S, \quad a_2 - S, \quad a_3 - S, \quad a_4 - S, \quad a_5 - S, \quad a_6 - S, \quad a_7 - S, \quad a_8 - S, \quad a_9 - S, \quad a_{10} - S, \quad a_{11} - S, \quad a_{12} - S, \quad a_{13} - S, \quad a_{14} - S, \\
    &a_{15} - S, \quad \text{and} \quad a_{16} - S.
\end{align*}
\]

The residual pixels are obtained by subtracting predicted pixels (\( S \)) with current sub block pixels then apply the integer transform, Gaussian pulse, quantization and inverse quantization, inverse transform to get reconstructed block.

The operation involved in DC prediction mode for intra frame is as shown in Figure 5.12.

![Figure 5.12 DC Intra prediction mode for intra frame](image)
5.8.4 Operation of Diagonal down left prediction mode (Mode-3)

For diagonal down left intra prediction mode, all the pixels of current sub block is predicted by a weighted average of the prediction samples A to M. Prediction equation generated for diagonal down left mode as follows.

\[ a_1 = ((A+B) + (B+C) +2) >> 2 = (A+2B+C+2) >> 2 = S \]
\[ a_2 = a_5 = ((C+D) + (B+C) +2) >> 2 = (B+2C+D+2) >> 2 = T \]
\[ a_3 = a_6 = a_9 = ((C+D) + (D+E) +2)>>2 = C+2D+E+2>>2 = U \]
\[ a_4 = a_7 = a_{10} = a_{13} = ((E+F) + (D+E) +2) >> 2 = (D+2E+F+2) >> 2 = V \]
\[ a_8 = a_{11} = a_{14} = ((E+F) + (F+G) +2) >> 2 = (E+2F+G+2) >> 2 = W \]
\[ a_{12} = a_{15} = ((G+H) + (H+H) +2) >> 2 = (G+3H+2) >> 2 = X \]
\[ a_{16} = ((G+H) + (F+G) +2)>>2 = (G+2G+H+2) >> 2 = Y \]

The operation involved in diagonal down left intra prediction mode for intra frame as shown in Figure 5.13.

A residual pixel is obtained by:
Prediction pixels of current sub block (a1 to a16) - Predicted pixels (STUVWXY)
\[ a_1 - S, a_2 - T, a_3 - U, a_4 - V, a_5 - T, a_6 - U, a_7 - V, a_8 - W, a_9 - Y, a_{10} - V, a_{11} - W, a_{12} - U, a_{13} - V, a_{14} - W, a_{15} - Y \text{ and } a_{16} - X \]

These residual pixels are processed through the process of 4x4 integer transform, generated Gaussian pulse, Quantization, encoder, inverse quantization and inverse transform, finally measuring the quality, bit rate and compression. The operation involved in diagonal downright intra prediction mode for intra frame as shown in Figure 5.13.
5.8.5 Operation of Diagonal downright prediction mode (Mode-4)

For diagonal downright intra prediction mode, all the pixels of current sub block is predicted by a weighted average of the prediction samples A to M. Prediction equation generated for diagonal downright mode intra prediction as follows.

\[
\begin{align*}
    a_1 &= a_6 = a_{11} = a_{16} = ((I+M) + (M+A) +2) >> 2 = (I+2M+A+2) >> 2 = S \\
    a_2 &= a_7 = a_{12} = ((M+A) + (A+B) +2) >> 2 = (M+2A+B+2) >> 2 = T \\
    a_3 &= a_8 = ((A+B) + (B+C) +2) >> 2 = (A+2B+C+2) >> 2 = U \\
    a_4 &= ((C+D) + (C+B) +2) >> 2 = (B+2C+D+2) >> 2 = V \\
    a_5 &= a_{10} = a_{15} = (J+2I+M+2) >> 2 = W \\
    a_{13} &= ((K+L) + (K+J) +2) >> 2 = (L+2K+J+2) >> 1 = X
\end{align*}
\]
\[ a_9 = a_{14} = ((K+J) + (I+K) + 2) \gg 2 = (K+2J+I+2) \gg 2 = Y \]

Residual pixels are obtained by

\[ a_1 - S, a_2 - T, a_3 - U, a_4 - V, a_5 - W, a_6 - S, a_7 - T, a_8 - U, a_9 - Y, a_{10} - W, a_{11} - S, a_{12} - T, a_{13} - X, a_{14} - Y, a_{15} - W \text{ and } a_{16} - S \]

Figure 5.14 Diagonal down Right intra prediction mode for Intra frame

These residual pixels are processed through the process of 4x4 integer transform, generated Gaussian pulse, Quantization, encoder, inverse quantization and inverse transform, finally measuring the quality, bit rate and compression. The operation involved in diagonal downright intra prediction mode for intra frame as shown in Figure 5.14.
5.8.6 Operation of vertical right prediction mode (Mode-5)

For vertical right intra prediction mode, all the pixels of current sub block is predicted by an average weighted of the prediction samples A to M.

Prediction equation generated for vertical right mode as follows

\[ a_{13} = ((K+J) + (K+I) +2) \gg 2 = (K+2J+I+2) \gg 2 = P_1 \]
\[ a_9 = ((J+I) + (I+M) +2) \gg 2 = (J+2I+M+2) \gg 2 = Q_1 \]
\[ a_5 = a_{14} = ((I+M) + (A+M) +2) \gg 2 = (I+2M+A+2) \gg 2 = R_1 \]
\[ a_6 = a_{15} = ((M+A) + (M+B) +2) \gg 2 = (M+2A+B+2) \gg 2 = S_1 \]
\[ a_7 = a_{16} = ((B+A) + (B+C) +2) \gg 2 = (A+2B+C+2) \gg 2 = T_1 \]
\[ a_1 = a_{10} = (M+A+1) \gg 1 = V_1 \]
\[ a_2 = a_{11} = (A+B+1) \gg 1 = W_1 \]
\[ a_3 = a_{12} = (B+C+1) \gg 1 = X_1 \]
\[ a_4 = (C+D+1) \gg 1 = Y_1 \]

\[ P_1 \ Q_1 \ R_1 \ S_1 \ T_1 \ U_1 \ V_1 \ W_1 \ X_1 \ \text{and} \ Y_1 \] is the predicted samples and \( a_1 \) to \( a_{16} \) are prediction samples of current sub block.

The operation involved in vertical Right intra prediction mode for intra frame as shown in Figure 5.15.

The residual samples are obtained by

\[ a_1-V_1, \ a_2-W_1, \ a_3-X_1, \ a_4-Y_1, \ a_5-R_1, \ a_6-S_1, \ a_7-T_1, \ a_8-U_1, \ a_9-Q_1, \ a_{10}-V_1, \ a_{11}-W_1, \ a_{12}-X_1, \ a_{13}-P_1, \ a_{14}-R_1, \ a_{15}-S_1 \ \text{and} \ a_{16}-T_1. \]

These residual pixels are processed through the process of 4x4 integer transform, generated Gaussian pulse, Quantization, encoder, inverse quantization and inverse transform, finally
measuring the quality, bit rate and compression. The operation involved in vertical right intra prediction mode for intra frame as shown in Figure 5.15.

**Figure 5.15** Vertical right intra prediction mode for Intra frame
5.8.7 Operation of Horizontal down prediction mode (Mode-6)

For Horizontal down intra prediction mode all the pixels of current sub block is predicted by a weighted average of the prediction samples A to M.

Prediction equation generated for Horizontal down mode as follows.

\[
\begin{align*}
a_1 &= a_7 = (I+M+1) >> 1 = P_{11} \\
a_2 &= a_8 = ((I+M)+(A+M)+2)>>2 = (I+2M+A+2) >> 2 = Q_{11} \\
a_3 &= ((M+A)+(A+B)+2)>>2 = (M+2A+B+2) >> 2 = R_{11} \\
a_4 &= ((B+A)+(C+B)+2)>>2 = (A+2B+C+2) >> 2 = S_{11} \\
a_5 &= a_{11} = (J+I+1) >> 1 = T_{11} \\
a_9 &= a_{15} = (K+J+1) >>1 = U_{11} \\
a_{13} &= (L+K+1) >>1 = V_{11} \\
a_{14} &= ((L+K)+(J+K)+2)>>2 = (L+2K+J+2) >> 2 = W_{11} \\
a_{10} &= a_{16} = ((K+J)+(J+I)+2)>>2 = (K+2J+I+2) >> 2 = X_{11} \\
a_6 &= a_{12} = ((J+I)+(M+I)+2)>>2 = (J+2I+M+2) >> 2 = Y_{11}
\end{align*}
\]

The residual pixel are obtained by subtract current sub block pixels (a_1 to a_{16} ) and predicted pixels (P_{11}, Q_{11}, R_{11}, S_{11}, T_{11}, U_{11}, V_{11}, W_{11}, X_{11}, Y_{11})

\[
\begin{align*}
a_1-P_{11}, a_2-Q_{11}, a_3-R_{11}, a_4-S_{11}, a_5-T_{11}, a_6-Y_{11}, a_7-P_{11}, a_8-Q_{11}, a_9-U_{11}, a_{10}-X_{11}, a_{11}-T_{11} \\
a_{12}-Y_{11}, a_{13}-V_{11}, a_{14}-W_{11}, a_{15}-U_{11}, a_{16}-X_{11}
\end{align*}
\]

These residual pixels are processed through the process of 4x4 integer transform, generated Gaussian pulse, Quantization, encoder, inverse quantization and inverse transform, finally measuring the quality, bit rate and compression.
5.8.8 Operation of Vertical left prediction mode (Mode-7)

For vertical left intra prediction mode all the pixels of current sub block is predicted by a weighted average of the prediction samples A to M.

Prediction equation generated for vertical left mode as follows.

\[ a_2 = a_9 = (B+C+1)>>1 = P_{111} \]
\[ a_3 = a_{10} = (C+D+1)>>1 = R_{111} \]
\[ a_4 = a_{11} = (D+E+1)>>1 = S_{111} \]
\[ a_{12} = (E+F+1)>>1 = T_{111} \]
\[ a_{16} = ((E+F)+(F+G)+2) >> 2 = (E+2F+G+2)>>2 = U_{111} \]
\[ a_8 = a_{15} = ((E + F) + (D + E) + 2) >> 2 = (D+2E+F+2)>>2 = V_{111} \]
\[ a_7 = a_{14} = ((C + D) + (D + E) + 2) >> 2 = (C+2D+E+2)>>2 = W_{111} \]
\[ a_6 = a_{13} = ((C + D) + (B + C) + 2) >> 2 = (B+2C+D+2)>>2 = X_{111} \]
\[ a_5 = ((A + B) + (B + C) + 2) >> 2 = (A+2B+C+2)>>2 = Y_{111} \]

The residual pixel are obtained by subtract current sub block pixels (a_1 to a_{16} ) and predicted pixels (P_{111} Q_{111} R_{111} S_{111} T_{111} U_{111} V_{111} W_{111} X_{111} Y_{111})

\[ a_1 - Q_{111}, a_2 - P_{111}, a_3 - R_{111}, a_4 - S_{111}, a_5 - Y_{111}, \]
\[ a_6 - X_{111}, a_7 - W_{111}, a_8 - V_{111}, a_9 - P_{111}, a_{10} - R_{111}, a_{11} - S_{111}, a_{12} - T_{111}, a_{13} - X_{111}, a_{14} - W_{111}, a_{15} - V_{111}, a_{16} - U_{111} \]

These residual pixels are processed through the process of 4x4 integer transform, generated Gaussian pulse, Quantization, encoder, inverse quantization and inverse transform, finally measuring the quality, bit rate and compression.
5.8.9 Operation of Horizontal Up prediction mode (Mode-8)

For Horizontal up intra prediction mode all the pixels of current sub block is predicted by a weighted average of the prediction samples A to M.

Prediction equation generated for Horizontal up mode as follows

\[ a_1 = (J+I+1) >> 1 = P_2 \]
\[ a_2 = ((I + J) + (J + K) + 2) >> 2 = (K+2J+I+2) >> 2 = Q_2 \]
\[ a_3 = a_5 = (K+J+1) >> 1 = R_2 \]
\[ a_4 = a_6 = ((J + K) + (K + L) + 2)) >> 2 = (L+2K+J+2) >> 2 = S_2 \]
\[ a_7 = a_9 = (L+K+1) >> 1 = T_2 \]
\[ a_8 = a_{10} = ((L + L) + (K + L) + 2)) >> 2 = (3L+J+2) >> 2 = U_2 \]
\[ a_{11} = a_{12} = a_{13} = a_{14} = a_{15} = a_{16} = V_2 \]

The residual pixel are obtained by subtract current sub block pixels (a_1 to a_{16}) and predicted pixels (P_2 Q_2 R_2 S_2 T_2 U_2 V_2)

\[ a_1 - P_2, \quad a_2 - Q_2, \quad a_3 - R_3, \quad a_4 - S_2, \quad a_5 - R_2, \quad a_6 - S_2, \quad a_7 - T_2, \quad a_8 - U_2, \quad a_9 - T_2, a_{10} - U_2, \quad a_{11} - V_2, a_{12} - V_2, a_{13} - V_2, a_{14} - V_2, a_{15} - V_2, a_{16} - V_2 \]

These residual pixels are processed through the process of 4x4 integer transform, generated Gaussian pulse, Quantization, encoder, inverse quantization and inverse transform, finally measuring the quality, bit rate and compression.
5.9 Algorithm of proposed method for Intra frame

The proposed research algorithm measures the performance parameters PSNR, bit rate, encode time and compression of the intra-frame coding. The encoding of the intra-frame consists of intra prediction, integer transform, Gaussian pulse quantization and entropy (CAVLC) inverse quantization, inverse transform. The first frame of input video is Intra frame which is coded by MATLAB and compare with Intra frame tested by JM18.6algorithm of AVC.

The present work is on intra frame coding in H.264/ Advanced Video Codec using Gaussian method for all nine Intra prediction modes was coded in MATLAB. Intra frame coding of H.264 involves various modules such as intra prediction, integer transform, quantization, Gaussian function, context adaptive variable length coding, inverse quantization and inverse quantization were coded and implemented in MATLAB and compared with recent version JM reference algorithm of H.264 Standard. Intra frame is successfully coded, tested and implemented for different CIF and QCIF yuv video sequences using MATLAB and simulation results of proposed method are compared with JM 18.6 reference algorithm. The simulation results of proposed method shows that consistent PSNR, reduced in bit rate, reduce in encoding time and high compression achieved.

5.9.1 Steps for Intra frame coding

- A yuv test video sequences dataset is considered as input
- Perform extracting intra frames and consider it as new input
- An Intra frame is process in terms of macro blocks. The size of each macro block is 16x16
- Each macro block (16x16) is further divided into 4x4 sub macro block
- Perform processing of preliminary sub- block without any prior reconstructed block
- Perform mathematical transformation, Gaussian multiplication, quantization, inverse quantization, inverse transform for reconstruction of preliminary sub block, and entropy encoding using CAVLC to get compression bit stream
• Perform subtraction of consecutive sub-block with prior reconstructed sub-block and extract residual blocks.
• Repeat (perform mathematical transformation, Gaussian multiplication, quantization, inverse quantization, inverse transform for reconstruction of preliminary sub block, and entropy encoding using CAVLC to get compression bit stream) for all the sub block of Intra frames.
• Evaluated the performance parameters such as PSNR to measure picture quality, bit rate, and encode time and compression of Intra frames.
• Compare the evaluated the performance parameters such as PSNR (picture quality), bit rate, and encode time and compression of proposed algorithm with JM 18.6 reference algorithm of AVC/H.264.

5.9.2 The detail steps are performed for Intra frame coding

The following are detail steps performed for intra frame coding to achieve research objectives outlined in third chapter.

• The input test yuv video sequences in QCIF (176x144) and CIF (352x288) format, which is open source, and widely used International standard test video sequences downloaded from the website: www.codersvoice.com
• Read Intra frame i.e. I-frame (extracted I-frames) from different standard test yuv video sequences.
• Intra frame are processed in the form of macro blocks; each macro block size is 16x16.
• Each macro block is further divided into a 4x4 sub macro blocks.
• A first 4x4 sub block of Intra frame is processed directly followed by forward path of encoder
• Obtained reconstruct first 4x4 sub block followed by inverse quantization and inverse integer transform.
• Obtained prediction block using prediction equations of 4x4 intra prediction modes based on previously reconstructed samples of reconstructed block.
- Obtained residual block \((X)\) by subtracting prediction block with current 4x4 sub block.
- Applied 4x4 integer transform for residual coefficients to get transformed coefficients i.e. \(W = A \times X \times A^T\), Where \(W\) is transformed coefficients, \(A\) is integer constant matrix and \(A^T\) transpose of integer constant matrix.
- Multiplied transformed coefficients \((W)\) with a generated Gaussian pulse equation \(G(v) = \exp(-v^2)\) at macro block level then Gaussian transformed coefficients are obtained which was in form of abstract of frequency domain samples without intermixing the information among the samples. The Gaussian transformed coefficients are \(GN = W \times G(v)\), for one Gaussian pulse. For two Gaussian pulse \(GN = W \times (G_1(v) \times G_2(v))\).
- Applied quantization operation for Gaussian transformed coefficients \((GN)\) to get quantized coefficients i.e. \(Z = \text{round} (GN \times (MF/2^{qbits}))\) Where MF is multiplication factor, after that quantized coefficients are scanned in zigzag manner and reordered.
- Applied Context based Adaptive Variable Length Coding (CAVLC) for reordered quantized coefficients, there are five steps are carried for quantized coefficients to get compressed bit stream i.e. (i) Encoded on coefficients and the trailing ones, (ii) Encoded the sign of each trailing ones, (iii) Encoded the level of non-zero coefficient, (iv) Encoded the total number of zeros before the last coefficient and (v) Encoded each run of zeros.
- Obtained reconstruction block, with using inverse quantization and inverse transform along with addition of prediction that is the following
- Applied inverse quantization operation for quantized coefficient i.e. \(W' = Z \times U \times 2^{\text{floor}(QP/6)}\) Where \(W'\) is the inverse quantized coefficients, \(Z\) is the quantized coefficients and \(U\) is rescaling parameters \((U = Qstep.PF.64)\).
- Applied inverse transform operation for inverse quantized coefficients \((W')\) i.e. \(X' = A_i^T \times W'^* A_i\) Where \(W'\) inverse quantized coefficients; \(A_i\) is a constant inverse integer matrix, \(A_i^T\) is a transpose constant inverse integer matrix, after inverse transform it get difference block i.e. \((X')\).
- Reconstructed block is obtained by adding prediction block to difference block \((X')\) i.e. Reconstructed block = Prediction block + Difference block \((X')\).
- Reconstruction block is serves as prediction for next sub block of intra frame.
• The above steps repeat for all the remaining sub blocks of intra frame.
• Finally measured performance parameters such as PSNR, bit rate, encode time and compression of Intra frame.
• The consistent PSNR is obtained by i.e. Constant PSNR or Consistent PSNR = Integer Transform (IT)/Integer Transform – Integer Transform *(Gaussian pulse)
• The bit rate, encode time; PSNR and compression are measured using the standard performance indices or metrics or equations.
• Compared the evaluated the performance parameters such as PSNR i.e. picture quality, bit rate, and encode time and compression of proposed algorithm with JM 18.6 reference algorithm of AVC/H.264 along with some of previous works of researchers.
• Plotted graphs such as rate-distortion, bit rate and compression of intra frames for different QP with different Gaussian values.
• For all 4x4 intra prediction modes, tabulated performance parameters PSNR, bit rate, encode time and compression ratio of intra frames under different QP and Gaussian values.

5.9.3 Operations of each stage of proposed for one sub block

Intermediate operations of each stage for one 4x4 sub macro block of intra frame with quantization and Gaussian value as follows. With Gaussian pulse, the operation of intermediate stage of proposed research for 4x4 sub macro block of Intra frame under quantization parameter 20, Gaussian value 0.1 with DC prediction mode (mode-2) are presented shown in below.

• Current 4x4 sub block of Intra frame

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>151</td>
<td>149</td>
<td>151</td>
<td>155</td>
</tr>
<tr>
<td>150</td>
<td>153</td>
<td>158</td>
<td>164</td>
</tr>
<tr>
<td>158</td>
<td>163</td>
<td>169</td>
<td>172</td>
</tr>
<tr>
<td>167</td>
<td>170</td>
<td>174</td>
<td>177</td>
</tr>
</tbody>
</table>
• Prediction block obtained using prediction equation of intra prediction mode

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>149</td>
<td>149</td>
<td>149</td>
<td>149</td>
<td></td>
</tr>
<tr>
<td>148</td>
<td>148</td>
<td>148</td>
<td>148</td>
<td></td>
</tr>
<tr>
<td>154</td>
<td>154</td>
<td>154</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>163</td>
<td>163</td>
<td>163</td>
<td>163</td>
<td></td>
</tr>
</tbody>
</table>

• Obtain residual block by subtraction of prediction block with current block

\[
\text{Residual block (X)} = \text{Current sub block} - \text{Predicted block}
\]

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>10</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>15</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>11</td>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>

• Residual coefficients are transformed using 4x4 integer transform

The residual values are transformed into transformed coefficients using 4x4 integer transform equation which is given by \( W = A \times X \times A^T \), where \( X \) is the input 4x4 residual block or data, \( W \) is the matrix of transformed coefficients, \( A \) and \( A^T \) transpose of \( A \) are constant modified DCT matrices i.e. contain integers.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>125</td>
<td>-101</td>
<td>7</td>
<td>-8</td>
</tr>
<tr>
<td>-65</td>
<td>29</td>
<td>17</td>
<td>2</td>
</tr>
<tr>
<td>-33</td>
<td>33</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>12</td>
<td>-4</td>
<td>6</td>
</tr>
</tbody>
</table>
Multiply the first row of integer matrix \((A)\) with each column of the residual values then it generate first row of products which is multiplied with each column of the transpose of integer matrix \((A^T)\) to generate the first row of integer transformed coefficients. Repeat the procedure for remaining rows of integer matrix with each column of residual and transpose matrix to get remaining transform coefficients.

- **Obtain the Gaussian coefficients** :

  Integer Transform coefficients (IT) * Gaussian pulse

**Effect Gaussian Pulse**

Multiplying the generated Gaussian pulse with integer transformed coefficients of macro block level which is in terms of 4x4 sub block then generate abstract frequency domain samples. It avoids intermix of the frequency domain information among the samples. This multiplication operation performs or scales the information content of the signal in a reversible way and avoids drastic reduction of quality of the picture and also improves functionality of quantizer. The degree of abstraction can controlled through a simple parameter—that is Gaussian scaling parameter. The resulting block of Gaussian coefficients are.

\[
\begin{pmatrix}
18 & -14 & 1 & -1 \\
-9 & 4 & 2 & 0 \\
-4 & 4 & 0 & 0 \\
0 & 1 & 0 & 0
\end{pmatrix}
\]

- **Obtain quantization coefficients**

Quantization reduces the precision of the coefficients according to a quantization parameter. AVC supports a 52 quantization step which is indexed by quantization parameter. The quantization steps control the trade off between bit rate and quality. The Gaussian coefficients are quantized using quantization process by the following equation.
\[ Z = \text{round} \left( \text{GN.} \left( \frac{\text{MF}}{2^{\text{qbits}}} \right) \right) \quad \text{Equation 5.15} \]

The equivalent of \( \frac{\text{MF}}{2^{\text{qbits}}} \) is \( \frac{\text{PF}}{\text{Qstep}} \). The multiplication parameter is specified in AVC. Multiplication parameter is divide by \( 2^{\text{qbits}} \), where \( \text{qbits} = 15 + \text{floor} \left( \frac{\text{QP}}{6} \right) \), which is multiplied with each transformed coefficients to generate quantized coefficients i.e. \( Z \) is the quantized coefficients.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>-14</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>-9</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>-4</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Quantized coefficients are encoded using Context Adaptive Variable Length coding to generate encoded bits i.e. compressed bits.**

- **Inverse Quantized coefficients**

The inverse quantization coefficients are obtained by the operation which is given as

\[ W' = Z.U.2^{\text{floor} \left( \frac{\text{QP}}{6} \right)} \quad \text{Equation 5.16} \]

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>120.06</td>
<td>-93.38</td>
<td>6.67</td>
<td>-6.67</td>
</tr>
<tr>
<td>-60.03</td>
<td>26.68</td>
<td>13.34</td>
<td>0</td>
</tr>
<tr>
<td>-26.68</td>
<td>26.68</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>6.67</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Inverse Transform coefficients**

Inverse transform coefficients are obtained by

\[ X' = A_i^* W' A_i \quad \text{Equation 5.17} \]
• **Reconstructed block**

Reconstructed block is obtained by adding prediction block to inverse transformed coefficients i.e.

\[
\text{Reconstructed block} = \text{Predicted block} + X'
\]

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>9</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>10</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

• **Reconstructed block is approximately current sub block**

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>149</td>
<td>149</td>
<td>150</td>
<td>154</td>
<td></td>
</tr>
<tr>
<td>148</td>
<td>151</td>
<td>157</td>
<td>163</td>
<td></td>
</tr>
<tr>
<td>156</td>
<td>161</td>
<td>168</td>
<td>173</td>
<td></td>
</tr>
<tr>
<td>165</td>
<td>170</td>
<td>173</td>
<td>176</td>
<td></td>
</tr>
</tbody>
</table>

After performing all steps of proposed method it observes that the reconstructed block is approximate equal to the current sub block. The above steps are performed for only one sub block. The remaining sub blocks of Intra frame are performed by repeating above steps.
5.10 Performance Parameters

The performance parameters of proposed algorithm are picture quality (PSNR), bit rate, encode time and compression. Each parameter discussed in subsequent section.

5.10.1 PSNR

Peak Signal to Noise Ratio abbreviated as PSNR. It is objective measure of video quality, which measures the difference between the original image and reconstructed images. It is defined as the ratio of the maximum possible signal power of the original image to the power of decompressed image. For an image sequence, high PSNR indicates high quality and low PSNR indicates low quality [117].

Mathematically PSNR is represented as

$$\text{PSNR} (\text{dB}) = 10 \times \log_{10} \frac{(2^n - 1)^2}{\text{MSE}} \quad \text{Equation 5.18}$$

Where \( n \) is number of bits per image sample, \((2^n - 1)^2\) is the highest possible signal value in the image or the peak value of an 8-bit image.

MSE is Mean square error; MSE is Mean Squared error between pixel values of original image and pixel values of reconstructed or distorted image.

MSE is calculated using the following equation

$$\text{MSE} = \frac{1}{M \times N} \sum \sum (I_o (i, j) - (I_o' (i, j))^2 \quad \text{Equation 5.19}$$

Where, \( I_o (i, j) \) pixel of original image and \( I_o' (i, j) \) pixel of distorted image

Video quality is depends on value of PSNR, higher the PSNR gives higher video quality and lower PSNR gives lower video quality. PSNR value greater than 30 dB is acceptable and PSNR value of 35 dB and above is implies that the reconstructed picture is indistinguishable from the original picture.
5.10.2 Bit rate

In proposed research work the following formula used for measuring the bit rate in bits per second. It can also be measured in terms in kilo bits per second (kbps) and mega bits per second (Mbps). The average bit rate per second is determined by number of bits in compressed bit streams, the frame rate i.e. number of frames per second and number of frames of video sequence.

The bit rate is measured in bits per second as follows

\[
\text{Bit rate} = \frac{(\text{Number of compressed bits}) \times (\text{Number of frames per second})}{(\text{Number of frames})}
\]

5.10.3 Compression Ratio

Compression is reducing the number of bits of the image; it is defined as the ratio of original image to compressed image. Compression efficiency is given in the form of a compression ratio. The compression depends on the picture contents of the original picture.

In the proposed research work compression ratio calculated by

\[
\text{Compression Ratio} = \frac{\text{original picture size/compressed picture size or Total size in bits of original input image}}{\text{Total size in bits of compressed bitstream}}
\]

For example, video contains a sequence of still images recorded with standard definition which uses a resolution of 720x576 with a frame rate of 25fps, 4:2:0 format and 8-bit colour depth takes \(720 \times 576 \times 25 \times 8 + 2 \times (360 \times 576 \times 25 \times 8) = 1.66\) Mbps. Using any compression method which is compressed to 0.8 Mbps, then compression ratio is 2.075.