Information Hiding in H.264 Compressed Video

AN INTERIM PROJECT REPORT UNDER THE GUIDANCE OF DR K. R. RAO

COURSE: EE5359 – MULTIMEDIA PROCESSING, SPRING 2014

SUBMISSION Date: 04/02/14

SUBMITTED BY

VISHNU VARDHAN REDDY MUKKU

Mav ID: 1000989621

Email: vishnu.vardhanreddymukku@mavs.uta.edu

DEPARTMENT OF ELECTRICAL ENGINEERING

UNIVERSITY OF TEXAS, ARLINGTON.
ACRONYMS:

AVC : Advanced Video Coding
AEU : Australian Education Union
CAVLC : Context-Adaptive Variable Length Coding
CABAC : Context-Adaptive Binary Arithmetic Coding
CB : Candidate Block
CU : Coding Unit
DCT : Discrete Cosine Transform
FMO : Flexible Macroblock Ordering
GOP : Group of Pictures
HB : Host Block
ITU : International Telecommunication Union
JVT : Joint Video Team
QCIF : Quarter Common Intermediate Format
LSB : Least Significant Bit
MB : Macro Block
MPM : Most Probable Mode
RDO : Rate Distortion Optimization
SAE : Sum of Absolute Error
SP Frame : Stored P Frame
SI Frame : Stored I Frame
# TABLE OF CONTENTS

1. Abstract .................................................................................................................4

2. H.264 video coding standard ......................................................................................4
   2.1. Introduction ........................................................................................................4
   2.2. Encoder and Decoder in H.264 .........................................................................4

3. Prediction in H.264 ....................................................................................................7
   3.1. Macroblock prediction types ...............................................................................7
   3.2. Intra Prediction ...................................................................................................8
   3.3. Intra Mode Decision ..........................................................................................9
   3.4. RDO (Rate Distortion Optimization) .................................................................11

4. Algorithm of Information hiding approach .............................................................12
   4.1. Embedding Conditions ......................................................................................12
   4.2. Watermark Mapping Rules ...............................................................................13

5. Watermark embedding and extraction process .........................................................15
   5.1. Watermark embedding process .........................................................................15
   5.2. Watermark detection process ...........................................................................15

6. Experimental Results ...............................................................................................17

7. References ...............................................................................................................16
1. Abstract

An information hiding algorithm based on intra-prediction modes and matrix coding is proposed for H.264/AVC video stream. It utilizes the block types and modes of intra-coded blocks to embed watermark. Intra - 4 × 4 coded blocks (4x4-blocks) are divided into groups and two watermark bits are mapped to every three 4×4-blocks by matrix coding to map between watermark bit and intra-prediction modes. Since only the mode of an 4×4-block is changed for every two watermark bits, it can guarantee a high PSNR and slight bitrate increase after watermark embedding. Moreover, embedding position template is utilized to select candidate 4×4-blocks for watermark embedding, which further enhances the security of watermark information. Experimental results on several test sequences demonstrate that the proposed approach can realize blind extraction with real-time performance.

2. H.264 Video Coding Standard

2.1. Introduction:

H.264/Advanced Video Coding (AVC) [1] is video coding standard of the ITU-T Video Coding Experts Group and the ISO/IEC Moving Picture Experts Group. The main goals of the H.264/AVC standard were to enhance compression performance and provision of a network-friendly video representation addressing conversational (video telephony) and non-conversational (storage, broadcast, or streaming) applications [2].

2.2 Encoder and Decoder in H.264:

In comparison to the previous standards [3], H.264 incorporates various new features to further improve video compression efficiency. Notably, these features include intra prediction in intra-frame, multiple frames reference capability, quarter-pixel interpolation, deblocking filtering, and flexible macroblock ordering (FMO). In general, H.264 divides the sequence of frames (i.e., images) into several groups of pictures (GOPs). These frames are labeled as I (intra), P (predicted), and B (bi directionally predicted) frames, depending on the order in which they appear.
Fig. 1: H.264 hybrid video encoder [4].

Fig 2: Video coding: source frames, encoded bitstream, decoded frames [5]
Fig 3: H.264 decoder block diagram [6]

Fig 4: Architecture of the post deblocking filter [7]
The hybrid encoding process and the decoder block diagrams of the H.264 video compression standard are shown in Fig. 1 and Fig. 3 respectively. Fig. 2 shows, a sequence of original video frames or fields is encoded into the H.264 format, a series of bits that represents the video in compressed form. The Architecture of the post deblocking filter is shown in Fig 4. The source part, each frame is divided into non overlapping blocks of uniform size (i.e., 16×16 pixels) called macroblocks, and these macroblocks are handled uniquely depending on their types. Each macroblock can be further divided into smaller blocks as shown in Fig. 5, with 4×4 being the smallest possible block size. These macroblocks are subjected to integer version discrete cosine transform (DCT) [25], quantization, and entropy coding. First, the pixel values in a macroblock are used in the DCT and quantization process. The outputs of the DCT and quantization process, i.e., the quantized DCT coefficients, undergo the dequantization and inverse DCT process for prediction and motion estimation purposes. In particular, the intra- and inter-prediction processes utilize these reconstructed pixel values to execute pixel value estimation and to make decisions on coding-mode. Ordinarily, rate distortion optimization (RDO) is utilized to choose the best operational point between inter- or intra-mode for coding each macroblock. The code control block in Fig. 1 represents an optimizer that regulates the selection of coding modes and block sizes [8]. It also controls the quantization parameter to achieve the targeted video bitrate. Finally, the results of the DCT and quantization process, prediction data, motion vectors, control data from RDO are sent for entropy coding.

![Fig 5: Block size type selection for intra- and inter-macroblock to embed information [4]](image)

The output of entropy coding is a series of compressed video contents in the binary stream preceded and/or inter-leaved with various predefined markers. Context based Adaptive Variable Length Coding (CAVLC) [25] and Context based Adaptive Binary Arithmetic Coding (CABAC) [23] are the two widely used entropy coding techniques. The combined bit stream is then transmitted and/or stored in various media. HEVC is latest to H.264 and the special issues on HEVC [30] [31] [32] are presented in several papers.

3. Prediction in H.264

3.1. Macroblock prediction types [9]

Fig. 6 shows the prediction sources for three macroblocks. An I-Macroblock, a P-Macroblock and a B-Macroblock. An I Macroblock (I MB) is predicted using intra prediction from neighboring samples in the current frame. A P Macroblock (P MB) is predicted from samples in a previously-coded frame which may be before or after the current picture in display order, i.e. a ‘past’ or a ‘future’ frame. Different rectangular sections (partitions) in a P MB may be predicted from different reference frames.
Each partition in a B Macroblock (B MB) is predicted from samples in one or two previously-coded frames, for example, one ‘past’ and one ‘future’ as shown in the figure 6.

One important aspect of digital watermarking is the capacity or amount of data that can be embedded into a host signal. Most data-hiding techniques exploit perceptual masking to optimize capacity. Clearly, a higher capacity is achieved considering the inherent properties of different media when applying perceptual masking [18]. SP and SI [19] frames are the picture types used in H.264 for error-resilience and random access.

3.2 INTRA PREDICTION

An intra (I) macroblock [9] is coded without referring to any data outside the current slice. I macroblocks may occur in any slice type. Every macroblock in an I slice is an I macroblock. I macroblocks are coded using intra prediction, i.e. prediction from previously-coded data in the same slice. For a typical block of luma or chroma samples, there is a relatively high correlation between samples in the block and samples that are immediately adjacent to the block. Intra prediction therefore uses samples from adjacent, previously coded blocks to predict the values in the current block. In intra mode [16], a prediction block $P$ is formed based on previously encoded and reconstructed blocks and is subtracted from the current block prior to encoding. For the luminance samples, $P$ is formed for each $4 \times 4$ block or for a $16 \times 16$ macroblock. In an intra macroblock, there are three choices of intra prediction block size for the luma component, namely $16 \times 16$, $8 \times 8$ or $4 \times 4$. A single prediction block is generated for each chroma component. Each prediction block is generated using one of a number of possible prediction modes.

The choice of intra prediction block size for the luma component, $16 \times 16$, $4 \times 4$ or $8 \times 8$ when available, tends to be a trade-off between (i) prediction efficiency and (ii) cost of signaling the prediction mode [9]. (a) Smaller blocks: A smaller prediction block size ($4 \times 4$) tends to give a more accurate prediction, i.e. the intra prediction for each block is a good match to the actual data in the block. This in turn means a smaller coded residual, so that fewer bits are required to code the quantized transform.
coefficients for the residual blocks. However, the choice of prediction for every 4×4 block must be signaled to the decoder, which means that more bits tend to be required to code the prediction choices. Intra prediction source samples for 4×4 or I8 block is shown in Fig 7.(b) Larger blocks: A larger prediction block size (16×16) tends to give a less accurate prediction, hence more residual data, but fewer bits are required to code the prediction choice itself. Intra prediction source samples for I16 block is shown in Fig.8.

3.3. INTRA MODE DECISION:

Intra prediction in H.264 exploits the spatial correlation between the adjacent macroblocks. In JVT, the current macroblock is predicted by adjacent pixels in the upper and the left macroblocks that are decoded earlier. Then, the residual between the current macroblock and its prediction is transformed, quantized and entropy coded. Roughly speaking, the smaller the differences, the fewer the coding bits are demanded for the current macroblock. To get a richer set of prediction patterns, H.264 offers 9 prediction modes for 4×4 luma blocks (as in the Fig 9) and 4 prediction modes for 16×16 luma blocks. For the chrominance components, there are 4 prediction modes applied to the two 8×8 chroma blocks (U and V). The encoder has to select the best combination of prediction modes for each macroblock to obtain the optimal RD performance [10]. For intra-4×4 prediction, each mode has its own prediction direction and the predicted pixels a–p are obtained from a weighted average of reference pixels A–M. For instance, mode 2 is the dc prediction where all pixels are predicted by \((A + B + C + D + I + J + K + L)/8\). Fig.10 shows an example to the 9 modes in 4×4 block prediction.

![Intra prediction source samples, 4×4or8×8 luma blocks [9]](image)

![Intra prediction source samples, chroma or 16×16 luma blocks [9]](image)

![4×4 intra prediction modes [11] in H.264](image)
Table 1: Intra prediction modes in H.264 for 4×4 blocks [5]

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode 0 (Vertical)</td>
<td>The upper samples A,B,C,D are extrapolated vertically.</td>
</tr>
<tr>
<td>Mode 1 (Horizontal)</td>
<td>The left samples I,J,K,L are extrapolated horizontally.</td>
</tr>
<tr>
<td>Mode 2 (DC)</td>
<td>All samples in P are predicted by the mean of samples A..D and I..L.</td>
</tr>
<tr>
<td>Mode 3 (Diagonal Down-Left)</td>
<td>The samples are interpolated at a 45° angle between lower-left and upper-right.</td>
</tr>
<tr>
<td>Mode 4 (Diagonal Down-Right)</td>
<td>The samples are extrapolated at a 45° angle down and to the right.</td>
</tr>
<tr>
<td>Mode 5 (Vertical-Left)</td>
<td>Extrapolation at an angle of approximately 26.6° to the left of vertical, i.e. width/height = 1/2.</td>
</tr>
<tr>
<td>Mode 6 (Horizontal-Down)</td>
<td>Extrapolation at an angle of approximately 26.6° below horizontal.</td>
</tr>
<tr>
<td>Mode 7 (Vertical-Right)</td>
<td>Extrapolation or interpolation at an angle of approximately 26.6° to the right of vertical.</td>
</tr>
<tr>
<td>Mode 8 (Horizontal-Up)</td>
<td>Interpolation at an angle of approximately 26.6° above horizontal.</td>
</tr>
</tbody>
</table>

Table 1 gives the description of all the 9 intra prediction modes for 4×4 blocks in H.264.

### 16×16 Luma Prediction Modes:

As an alternative to the 4×4 luma modes described above, the entire 16×16 luma component of a macroblock may be predicted in one operation. Four modes are available, shown in Fig 11.
Mode 0 (vertical) Extrapolation from upper samples (H)
Mode 1 (horizontal) Extrapolation from left samples (V)
Mode 2 (DC) Mean of upper and left-hand samples (H+V).
Mode 4 (Plane) A linear ‘plane’ function is fitted to the upper and left-hand samples H and V.
This works well in areas of smoothly-varying luminance.

![Intra 16x16 prediction modes](image)

Fig 11: Intra 16×16 prediction modes [5] in H.264

Different algorithms are proposed for the intra prediction mode selection. Some of those techniques are RDO (rate distortion optimization) etc.

3.4. RDO (Rate Distortion Optimization):

The RDO method [12] searches, for each 4×4-block, the best mode with the metric of minimum rate distortion cost, and then $\text{OPT}_X$ is adopted as the prediction mode of $X$, named $M_X$. In addition to the rich set of intra-prediction modes, the correlation of spatially adjacent blocks is also exploited by H.264/AVC encoder to encode the prediction modes more efficiently. As illustrated in Fig 12, the most probable mode $\text{MPM}_X$ for the current 4×4-block is computed based on the prediction modes of its top block $A$ and left block $B$, which are denoted as $M_A$ and $M_B$, respectively, by $\text{MPM}_X = \min\{M_A, M_B\}$. The encoder sends a flag ($F$) for each 4×4-block to indicate whether $\text{MPM}_X$ is adopted as $M_X$ by

$$
F = 
\begin{cases} 
1; & \text{OPT}_X = \text{MPM}_X \\
0; & \text{OPT}_X \neq \text{MPM}_X
\end{cases}
$$

![Fig 12: 4x4-block X and its adjacent 4x4-blocks A and B](image)

Fig 12: 4×4-block $X$ and its adjacent 4×4-blocks $A$ and $B$ [12] in H.264
If the flag is 1, \( \text{MPM}_X \) is used and this time \( \text{OPT}_X = \text{MPM}_X \); Otherwise, another parameter \( \text{REM}_X \) is sent to indicate \( M_X \) and then it satisfies the equation \( M_X = \text{OPT}_X \) by

\[
\text{REM}_X = \begin{cases} 
\text{OPT}_X & ; \quad \text{OPT}_X < \text{MPM}_X \\
\text{OPT}_X - 1 & ; \quad \text{OPT}_X > \text{MPM}_X 
\end{cases}
\]

In this way, only 8 values (0–7) of \( \text{REM}_X \) are adequate to signal all possible modes of current 4×4-blocks.

Since intra-prediction is exploited to reduce spatial redundancy for compression purpose, the change of intra-prediction modes will cause bitrate increase.

---

4. **ALGORITHM OF INFORMATION HIDING APPROACH**

This algorithm hides secret data into qualified 4×4-blocks of I frames by modulating their prediction modes according to the mapping rules between these modes and binary watermark bits (w). Before the introduction of information embedding and extraction process, requirements for those host 4×4-blocks are specified and then the mapping rules are defined [12]. Fig 13 shows the process of embedding and extracting data using the H.264 codec. LSB matching [22] is the simplest information hiding technique among all.

![Diagram of information hiding process](image_url)

**Fig 13:** The process of embedding and extracting data using the H.264 codec [15]
4.1. Embedding Conditions

The luminance component of macroblocks in an I-frame is intra coded in either 16 × 16 or 4 × 4 intra-prediction modes. Each 4 × 4 block of residual data is transformed by an integer transform after intra-prediction. If the macroblock is coded in the 16 × 16 intra-prediction mode, the dc coefficients of all 4 × 4 blocks are transformed by a 4 × 4 Hadamard transform after the 4 × 4 integer transform [20] to further de-correlate these coefficients. In this approach the watermark is only embedded into the 4×4-blocks. The intra-coded I16-blocks, i.e., macroblocks, are not considered for watermark embedding. The two main reasons [12] are summarized as following: First, the 16 × 16 intra-prediction mode is used for smooth regions of the frame, so watermark embedding in these regions will easily leads to visible artifacts. Second, the extra Hadamard transform for these macroblocks further de-correlates the dc coefficients, so many of these coefficients are zero, which makes I16-block hard to be embedded with watermark information. Here, host block (HB) denotes the 4×4-block to embed watermark w, which satisfies the following two requirements at the same time: first, the 4×4-block X must be a candidate block (CB), which means the block having a flag F_X = 0. Second, the prediction modes of all 4×4-blocks within a 16 × 16 macroblock should be diverse. If more than 8 4×4-blocks are of the same prediction modes and are within two 8 × 8 sub-blocks, these 4×4-blocks will not be embedded with watermark information.

4.2. Watermark Mapping Rules

To decrease the bitrate increase and visual distortion by watermark embedding, the changes of intra-prediction modes should be as less as possible. However, the minimum changes of intra prediction modes still should be at the sacrifice of embedding capacity. In the work by Hu et al. [13], the optimal mode for 4×4-block is changed to sup-optimal mode to map watermark information. Except for the prediction mode of current 4×4-block, the other 8 candidate modes are divided into two groups. However, due to the diversities of video textures, the joint probabilities of optimal modes and sub-optical modes are small, which makes its mapping rules quite complex and not effective. Therefore, matrix coding is motivated to be introduced, so as to improve the mapping rules between the intra-prediction modes and the watermark bit [14]. For every three host 4×4-blocks, which meets the requirements for watermark embedding, they are remarked as a group. For matrix coding based mapping rules, every two watermark bits are modulated to the prediction modes of these three 4×4-blocks, thus only one 4×4-block is necessary to change its prediction mode for embedding two-bit watermark information. Compared with Hus work, this kind of mapping rules, which is detailed as following, can guarantee a higher PSNR and slighter bitrate increase after watermark embedding.
Table 2: The mapping rules for watermark embedding [12].

<table>
<thead>
<tr>
<th>Watermark to be embedded</th>
<th>Intra-prediction modes in a group</th>
<th>The 16-block to change its prediction mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>w = 00</td>
<td>Q1 ⊕ Q2 = 0, Q2 ⊕ Q3 = 0</td>
<td>No change</td>
</tr>
<tr>
<td></td>
<td>Q1 ⊕ Q2 = 1, Q2 ⊕ Q3 = 0</td>
<td>The first 4×4-block</td>
</tr>
<tr>
<td></td>
<td>Q1 ⊕ Q2 = 0, Q2 ⊕ Q3 = 0</td>
<td>The second 4×4-block</td>
</tr>
<tr>
<td>w = 01</td>
<td>Q1 ⊕ Q2 = 0, Q2 ⊕ Q3 = 0</td>
<td>The first 4×4-block</td>
</tr>
<tr>
<td></td>
<td>Q1 ⊕ Q2 = 1, Q2 ⊕ Q3 = 0</td>
<td>The second 4×4-block</td>
</tr>
<tr>
<td></td>
<td>Q1 ⊕ Q2 = 0, Q2 ⊕ Q3 = 0</td>
<td>No change</td>
</tr>
<tr>
<td>w = 10</td>
<td>Q1 ⊕ Q2 = 0, Q2 ⊕ Q3 = 0</td>
<td>The third 4×4-block</td>
</tr>
<tr>
<td></td>
<td>Q1 ⊕ Q2 = 1, Q2 ⊕ Q3 = 0</td>
<td>The first 4×4-block</td>
</tr>
<tr>
<td>w = 11</td>
<td>Q1 ⊕ Q2 = 0, Q2 ⊕ Q3 = 0</td>
<td>The second 4×4-block</td>
</tr>
<tr>
<td></td>
<td>Q1 ⊕ Q2 = 1, Q2 ⊕ Q3 = 0</td>
<td>The first 4×4-block</td>
</tr>
</tbody>
</table>

The 9 possible modes of 4×4-block are divided into two categories. The first category is the odd mode (1, 3, 5, 7), which is mapped to bit “1”. The second category is the even mode (0, 2, 4, 6, 8), which is mapped to bit “0”. Let Q1, Q2 and Q3 represent the corresponding bit information for the three 4×4-blocks in a group, and be the logical XOR operation. The mapping rules are defined in Table 2. When the intra-prediction mode should be changed, it is chosen by classical rate-distortion optimization (RDO) [13]. For example, when w = 00, Q1 = 0, Q2 = 1 and Q3 = 1, the modes of the three 4×4-blocks in a group satisfy the following equations:

\[
\begin{align*}
Q1 \oplus Q2 &= 1, \\
Q2 \oplus Q3 &= 0
\end{align*}
\]

According to the mapping rules defined in Table 2, the first 4×4-block in a group should change its prediction modes to map the watermark information. In other words, the mode should be changed from even mode to odd mode. Therefore, the sub-optimal mode for the first 4×4-block should be chosen from the odd mode (1, 3, 5 or 7) by RDO optimization.

To guarantee the security of hidden message, one possible approach is to pre-process [21] the watermark by encryption and scrambling. In addition, embedding position template, which refers to the selection of 4×4-blocks within a macroblock, is used to further enhance the security in this paper. In the design of mapping rules for watermark embedding, the candidates 4×4-blocks are divided into groups. Every group has three 4×4-blocks. Since there are 16 4×4-blocks within a macroblock, we can choose 3, 6, 9, 12 or 15 4×4-blocks from them to form groups for watermark embedding. Obviously, the number of 4×4-blocks chosen from every macroblock is closely related with the capacity of watermark embedding. In this paper, the maximum embedding density is 3/4. In other words, 12 4×4-blocks from 16 4×4-blocks are selected for watermark embedding, so there are 1820 kinds of embedding position templates. As illustrated in Fig.14(a) is the 4×4-blocks within a macroblock, which are numbered from 1 to 16 line by line. (b–d) are three typical embedding position templates. The selection of embedding position templates can be controlled by private key. For the grouping of 4×4-blocks within a template, the 4×4-blocks are divided into groups in raster order. If fact, the maximum
embedding density and the selection of embedding position templates are quite flexible. Therefore, it will guarantee the security of watermark information.

Fig 14: Embedding position templates [12]

5. Watermarking Embedding and Extraction process:

5.1. Watermark embedding process

The block diagram of information hiding scheme is illustrated in fig.13. Based on the prerequisite to host blocks as well as the mapping rule presented above, the proposed watermark embedding algorithm [12] can be implemented as following:

1. Encrypt and permute the watermark information.
2. Prepare two embedding bits w.
3. Select the embedding position template by private key.
4. For the three candidate 4×4-blocks, modify MX based on w and the mapping rules. If no change of intra-prediction modes in needed for these three 4×4-blocks, go to step 2.
5. Compute the new prediction mode of the 4×4-block to change its prediction mode, and re-encode it with the new prediction mode.
6. Return to step 2 until all the watermark information are embedded.
5.2. Watermark detection process

The retrieval process corresponds to the dual of the embedding process. Block diagram of Watermark detection process is illustrated in Fig. 15. The main steps are as following:

1. Partially decode the watermarked H.264/AVC video stream to get the prediction modes of all the 4×4-blocks.
2. Determine the embedding position templates with the private key.
3. According to the embedding position templates, choose all the watermarked 4×4-blocks.
4. For every group in the watermarked 4×4-blocks, recover the embedded watermark bit by the mapping rules.
5. Repeat step 4 until all the groups are finished.
6. Resemble the recovered watermark information, and then anti-scrambling and decryption of the recovered watermark information, thus the original watermark information can be obtained.

Obviously, the retrieval of hidden information is simple and fast. It can be accomplished without the original media and without complete decoding of input video stream. Only the prediction modes of those intra-coded 4×4-blocks by partial decoding are enough to detect the hided watermark information.
7. Experimental Results

Test sequences used:

<table>
<thead>
<tr>
<th>Test Sequence</th>
<th>Resolution</th>
<th>Frame Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>grandma_qcif</td>
<td>176 x 144</td>
<td>30</td>
</tr>
<tr>
<td>news_cif</td>
<td>352 x 288</td>
<td>24</td>
</tr>
<tr>
<td>silent_qcif</td>
<td>176 x 144</td>
<td>30</td>
</tr>
<tr>
<td>bridge-close_cif</td>
<td>352 x 288</td>
<td>30</td>
</tr>
</tbody>
</table>

1. grandma_qcif.yuv

2. news_qcif.yuv

3. silent_qcif.yuv

4. bridge-close_qcif.yuv

Fig 16: Test sequences used
Configuration of HM 13.0

Main all-intra profile settings

IntraPeriod : 1 # Period of I-Frame (-1 = only first)

GOPSize : 1 # GOP Size (number of B slice = GOPSize-1)

QP : 22 # Quantization parameter(0-51) (22, 27, 32 or 37 is used at a time)

Command line parameters for using HM 13.0 encoder:

TAppEncoder [-h] [-c config.cfg] [--parameter=value]

Options:

- h Prints parameter usage
- c Defines configuration file to use. Multiple configuration files may be used with repeated –c options.
--parameter=value Assigns value to a given parameter.

Sample command line parameters for HM 13.0 encoder:

C:\H.264JM1\bin>lencod.exe -f encoder.cfg -p inputfile"bridge_close_qcif" -p SourceWidth="176" SourceHeight="144"

RESULTS:

<table>
<thead>
<tr>
<th></th>
<th>HBQ (Hidden Bit Quantity)</th>
<th>BRI in % (Bit Rate Increase)</th>
<th>PSNRI in dB (PSNR Increase)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Silent</td>
<td>2231</td>
<td>0.4</td>
<td>-0.03</td>
</tr>
<tr>
<td></td>
<td>6648</td>
<td>2.1</td>
<td>-0.03</td>
</tr>
<tr>
<td></td>
<td>15561</td>
<td>4.3</td>
<td>-0.05</td>
</tr>
<tr>
<td></td>
<td>2231</td>
<td>0.7</td>
<td>-0.02</td>
</tr>
<tr>
<td></td>
<td>6648</td>
<td>2.2</td>
<td>-0.04</td>
</tr>
<tr>
<td></td>
<td>15561</td>
<td>4.1</td>
<td>-0.07</td>
</tr>
<tr>
<td></td>
<td>2231</td>
<td>1.5</td>
<td>-0.02</td>
</tr>
<tr>
<td></td>
<td>6648</td>
<td>3.3</td>
<td>-0.01</td>
</tr>
<tr>
<td></td>
<td>15561</td>
<td>5.9</td>
<td>-0.01</td>
</tr>
<tr>
<td></td>
<td>2231</td>
<td>0.9</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>6648</td>
<td>1.9</td>
<td>-0.03</td>
</tr>
<tr>
<td></td>
<td>15561</td>
<td>3.8</td>
<td>-0.04</td>
</tr>
</tbody>
</table>
7. References


