H.264 / MPEG 4 AVC Decoder Implementation

Fanyu Ran[a], Yang Zhou[b], and Yifei Zhou[c]

[a]Student No.8657223, University of Ottawa
[b]Student No.8657223, University of Ottawa
[c]Student No.8657223, University of Ottawa

ABSTRACT

This document is prepared using LaTeX2e[1] and shows the desired format and appearance of a manuscript prepared for the Proceedings of the SPIE.[*] It contains general formatting instructions and hints about how to use LaTeX. The LaTeX source file that produced this document, `article.tex` (Version 3.4), provides a template, used in conjunction with `spie.cls` (Version 3.4). These files are available on the Internet at https://www.overleaf.com. The font used throughout is the LaTeX default font, Computer Modern Roman, which is equivalent to the Times Roman font available on many systems.

Keywords: Manuscript format, template, SPIE Proceedings, LaTeX

1. INTRODUCTION

Begin the Introduction below the Keywords. The manuscript should not have headers, footers, or page numbers. It should be in a one-column format. References are often noted in the text and cited at the end of the paper.

Table 1. Fonts sizes to be used for various parts of the manuscript. Table captions should be centered above the table. When the caption is too long to fit on one line, it should be justified to the right and left margins of the body of the text.

Article title	16 pt., bold, centered
Author names and affiliations	12 pt., normal, centered
Keywords	10 pt., normal, left justified
Abstract Title	11 pt., bold, centered
Abstract body text	10 pt., normal, justified
Section heading	11 pt., bold, centered (all caps)
Subsection heading	11 pt., bold, left justified
Sub-subsection heading	10 pt., bold, left justified
Normal text	10 pt., normal, justified
Figure and table captions	9 pt., normal
Footnote	9 pt., normal
Reference Heading	11 pt., bold, centered
Reference Listing	10 pt., normal, justified

Further author information:
Fanyu Ran: E-mail: aaa@tbk2.edu
Yang Zhou: E-mail: bba@cmp.com
Yifei Zhou: E-mail: bba@cmp.com

[*]The basic format was developed in 1995 by Rick Hermann (SPIE) and Ken Hanson (Los Alamos National Lab.).

Table 2. Margins and print area specifications.

Margin	A4	Letter
Top margin	2.54 cm	1.0 in.
Bottom margin	4.94 cm	1.25 in.
Left, right margin	1.925 cm	.875 in.
Printable area	17.15 x 22.23 cm	6.75 x 8.75 in.

LaTeX margins are related to the document's paper size. The paper size is by default set to USA letter paper. To format a document for A4 paper, the first line of this LaTeX source file should be changed to `\documentclass[a4paper]{spie}`.

Authors are encouraged to follow the principles of sound technical writing, as described in Refs. 2 and 3, for example. Many aspects of technical writing are addressed in the *AIP Style Manual*, published by the American Institute of Physics. It is available on line at `https://publishing.aip.org/authors`. A spelling checker is helpful for finding misspelled words.

An author may use this LaTeX source file as a template by substituting his/her own text in each field. This document is not meant to be a complete guide on how to use LaTeX. For that, please see the list of references at `http://latex-project.org/guides/` and for an online introduction to LaTeX please see 4.

2. PYTHON PROGRAMMING LANGUAGE OVERVIEW

We chose Python as the main programming language to implement this project.

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java.

Although programs written in C/C++ mostly have significantly better performence then ones in Python, they have to pay for verbosity as tradeoff. Due to the statement from Python's official website, Python code is typically 5-10 times shorter than equivalent C++ code.

The dynamic property and modern syntax of Python can eliminate much more noise in the code, making the idea behind the program more clear. For example, to calculate Fibbonacci sequence in plain iterative method, we can implement it as below:

C:

```
long long int fibb(int n) {
    int fnow = 0, fnext = 1, tempf;
    while(--n>0){
        tempf = fnow + fnext;
        fnow = fnext;
        fnext = tempf;
        }
        return fnext;
}
```

Python:

```
def fibIter(n):
    if n < 2:
        return n
    fibPrev = 1
    fib = 1
    for num in xrange(2, n):
        fibPrev, fib = fib, fib + fibPrev
    return fib
```

As a project which aims to be a concept prototype, we care more about readability and productivity instead of performence. Python works well for this target.

3. H.264 DECODING OVERVIEW

The software architecture design is guided by the syntax of H.264 bitstream which is shown as below.

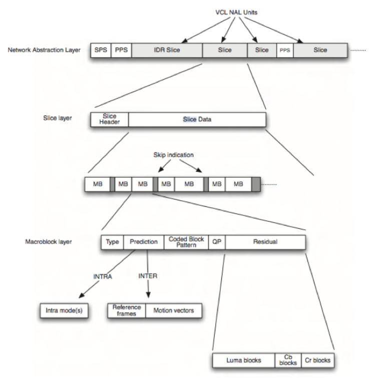

Figure 1. Figure captions are used to describe

The project is implemented mainly in object oriented paradigm. Primary concepts of H.264 are abstracted as objects. A simplified Unified Modeling Language(UML) diagram of the software architecture is shown as below:

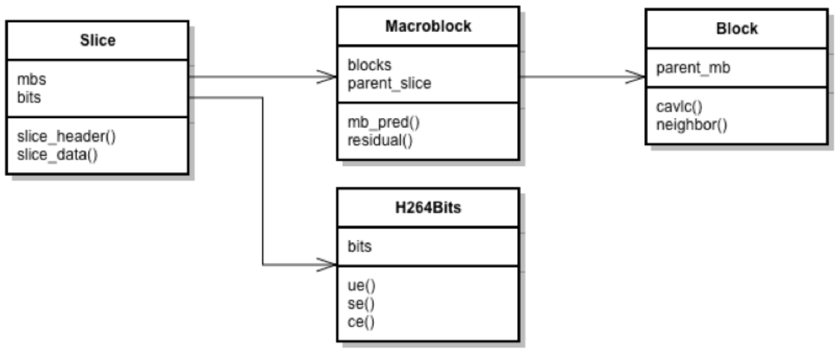

Figure 2. Figure captions are used to describe

The functionalities of these objects are described here.

3.1 H264Bits

H264Bits object stores raw bitstream data and entropy decoding methods. Each slice object contains a reference to the singleton H264Bits object, so the binary data can be decoded and transfered to its associated object.

3.2 SPS

SPS object contains sequence parameter set related parsing logic and result. The parsing methods read parameters from bitstream and store the results as instance variables of the object. So that these parameters can be shared later. There's usually only one SPS object in a sequence.

3.3 PPS

PPS object contains picture parameter set related parsing logic and result. This object has behavior similar to SPS except that there can be multiple PPS objects in a sequence.

3.4 Slice

Slice object contains slice header parsing logic, slice related parameters and macroblock objects. Slice-related paramters from slice header segments are parsed and stored here. Arbitrary Macroblock objects are contained in this object which is identical to H.264's layer logic.

3.5 Macroblock

Macroblock object contains macroblock parsing logic and Luma/Chroma block objects Macroblock-related parameters are decoded and CAVLC entropy decoding process is invoked here. The entropy decoded results are stored as Block objects inside this object.

3.6 Block

Block object contains Luma/Chroma block's transform coefficients and related decoding methods. The object also has block id related information stored in it.

Each object also contains its parent object pointer to share data between each other. So accessing sequence parameter set from a luma block object can be done like this:

```
one_luma_block.mb.slice.sps.some_parameter
```

There are another two modules for Intra frame decoding: idct and intra_pred. These modules are collections of functions for inverse DCT (and dequantization) and intra prediction.

3.7 IDCT

This module contains amount of functions to perform dequantization and inverse DCT on block coefficients levels. Related argument tables are also stored here as module attributes.

3.8 Intra Prediction

This module contains amount of functions to generate intra prediction for blocks.

4. SOFTWARE DESIGN OVERVIEW

The software architecture design is guided by the syntax of H.264 bitstream which is shown as below.

The project is implemented mainly in object oriented paradigm. Primary concepts of H.264 are abstracted as objects. A simplified Unified Modeling Language(UML) diagram of the software architecture is shown as below:

The functionalities of these objects are described here.

4.1 H264Bits

H264Bits object stores raw bitstream data and entropy decoding methods. Each slice object contains a reference to the singleton H264Bits object, so the binary data can be decoded and transfered to its associated object.

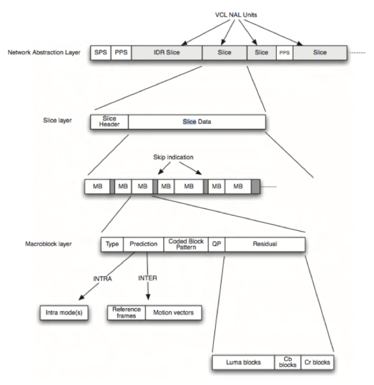

Figure 3. Figure captions are used to describe

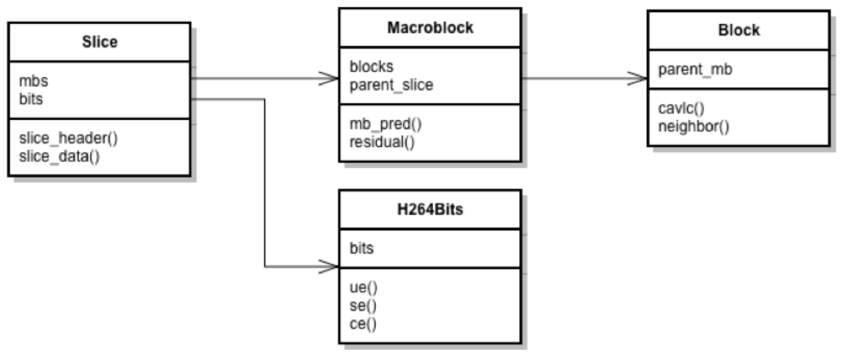

Figure 4. Figure captions are used to describe

4.2 SPS

SPS object contains sequence parameter set related parsing logic and result. The parsing methods read parameters from bitstream and store the results as instance variables of the object. So that these parameters can be shared later. There's usually only one SPS object in a sequence.

4.3 PPS

PPS object contains picture parameter set related parsing logic and result. This object has behavior similar to SPS except that there can be multiple PPS objects in a sequence.

4.4 Slice

Slice object contains slice header parsing logic, slice related parameters and macroblock objects. Slice-related paramters from slice header segments are parsed and stored here. Arbitrary Macroblock objects are contained in this object which is identical to H.264's layer logic.

4.5 Macroblock

Macroblock object contains macroblock parsing logic and Luma/Chroma block objects Macroblock-related parameters are decoded and CAVLC entropy decoding process is invoked here. The entropy decoded results are stored as Block objects inside this object.

4.6 Block

Block object contains Luma/Chroma block's transform coefficients and related decoding methods. The object also has block id related information stored in it.

Each object also contains its parent object pointer to share data between each other. So accessing sequence parameter set from a luma block object can be done like this:

```
one_luma_block.mb.slice.sps.some_parameter
```

There are another two modules for Intra frame decoding: idct and intra_pred. These modules are collections of functions for inverse DCT (and dequantization) and intra prediction.

4.7 IDCT

This module contains amount of functions to perform dequantization and inverse DCT on block coefficients levels. Related argument tables are also stored here as module attributes.

4.8 Intra Prediction

This module contains amount of functions to generate intra prediction for blocks.

5. H.264 AND IMPLEMENTATION DETAILS

Video and audio files can be included for publication. See Tab. **??** for the specifications for the mulitimedia files. Use a screenshot or another .jpg illustration for placement in the text. Use the file name to begin the caption. The text of the caption must end with the text "http://dx.doi.org/doi.number.goes.here" which tells the SPIE editor where to insert the hyperlink in the digital version of the manuscript.

Here is a sample illustration and caption for a multimedia file:

6. RESULT AND DESIGN EXPLORATION

Begin the Introduction below the Keywords. The manuscript should not have headers, footers, or page numbers. It should be in a one-column format. References are often noted in the text and cited at the end of the paper.

LaTeX margins are related to the document's paper size. The paper size is by default set to USA letter paper. To format a document for A4 paper, the first line of this LaTeX source file should be changed to `\documentclass[a4paper]{spie}`.

Authors are encouraged to follow the principles of sound technical writing, as described in Refs. 2 and 3, for example. Many aspects of technical writing are addressed in the *AIP Style Manual*, published by the American Institute of Physics. It is available on line at `https://publishing.aip.org/authors`. A spelling checker is helpful for finding misspelled words.

An author may use this LaTeX source file as a template by substituting his/her own text in each field. This document is not meant to be a complete guide on how to use LaTeX. For that, please see the list of references at `http://latex-project.org/guides/` and for an online introduction to LaTeX please see 4.

APPENDIX A. MISCELLANEOUS FORMATTING DETAILS

It is often useful to refer back (or forward) to other sections in the article. Such references are made by section number. When a section reference starts a sentence, Section is spelled out; otherwise use its abbreviation, for example, "In Sec. 2 we showed..." or "Section 2.1 contained a description...". References to figures, tables, and theorems are handled the same way.

Table 3. Fonts sizes to be used for various parts of the manuscript. Table captions should be centered above the table. When the caption is too long to fit on one line, it should be justified to the right and left margins of the body of the text.

Article title	16 pt., bold, centered
Author names and affiliations	12 pt., normal, centered
Keywords	10 pt., normal, left justified
Abstract Title	11 pt., bold, centered
Abstract body text	10 pt., normal, justified
Section heading	11 pt., bold, centered (all caps)
Subsection heading	11 pt., bold, left justified
Sub-subsection heading	10 pt., bold, left justified
Normal text	10 pt., normal, justified
Figure and table captions	9 pt., normal
Footnote	9 pt., normal
Reference Heading	11 pt., bold, centered
Reference Listing	10 pt., normal, justified

Table 4. Margins and print area specifications.

Margin	A4	Letter
Top margin	2.54 cm	1.0 in.
Bottom margin	4.94 cm	1.25 in.
Left, right margin	1.925 cm	.875 in.
Printable area	17.15 x 22.23 cm	6.75 x 8.75 in.

A.1 Formatting Equations

Equations may appear in line with the text, if they are simple, short, and not of major importance; e.g., $\beta = b/r$. Important equations appear on their own line. Such equations are centered. For example, "The expression for the field of view is

$$2a = \frac{(b+1)}{3c}\,, \tag{1}$$

where a is the ..." Principal equations are numbered, with the equation number placed within parentheses and right justified.

Equations are considered to be part of a sentence and should be punctuated accordingly. In the above example, a comma follows the equation because the next line is a subordinate clause. If the equation ends the sentence, a period should follow the equation. The line following an equation should not be indented unless it is meant to start a new paragraph. Indentation after an equation is avoided in LaTeX by not leaving a blank line between the equation and the subsequent text.

References to equations include the equation number in parentheses, for example, "Equation (1) shows ..." or "Combining Eqs. (2) and (3), we obtain..." Using a tilde in the LaTeX source file between two characters avoids unwanted line breaks.

A.2 Formatting Theorems

To include theorems in a formal way, the theorem identification should appear in a 10-point, bold font, left justified and followed by a period. The text of the theorem continues on the same line in normal, 10-point font. For example,

Theorem 1. For any unbiased estimator...

Formal statements of lemmas and algorithms receive a similar treatment.

ACKNOWLEDGMENTS

This unnumbered section is used to identify those who have aided the authors in understanding or accomplishing the work presented and to acknowledge sources of funding.

REFERENCES

[1] Lamport, L., [*LaTeX: A Document Preparation System*], Addison-Wesley, Reading, Mass. (1994).

[2] Alred, G. J., Brusaw, C. T., and Oliu, W. E., [*Handbook of Technical Writing*], St. Martin's, New York (2015 (eleventh edition)).

[3] Perelman, L. C., Paradis, J., and Barrett, E., [*Mayfield Handbook of Technical and Scientific Writing*], Mountain View, Mayfield (April 1997). http://mit.imoat.net/handbook/.

[4] Lees-Miller, J. D., "Free and Interactive Online Introduction to LaTeX." Overleaf, 26 February 2015 https://www.overleaf.com/latex/learn/free-online-introduction-to-latex-part-1. (Accessed: 15 September 2015).

[5] Goossens, M., Mittelbach, F., and Rahtz, S., [*The LaTeX Companion*], Addison-Wesley, Reading, Mass. (1997).

[6] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E., "Equations of state calculations by fast computing machine," *J. Chem. Phys.* **21**, 1087–1091 (1953).

[7] Gull, S. F., "Developments in maximum-entropy data analysis," in [*Maximum Entropy and Bayesian Methods*], Skilling, J., ed., 53–71, Kluwer Academic, Dordrecht (1989).

[8] Hanson, K. M., "Introduction to Bayesian image analysis," in [*Medical Imaging: Image Processing*], Loew, M. H., ed., *Proc. SPIE* **1898**, 716–731 (1993).