VLSI Architecture of Intra-Prediction and SAO Estimation in HEVC and its Extension to Compressed Sensing

July 2018

Jianbin ZHOU

周　剣斌

VLSI Architecture of Intra-Prediction and SAO Estimation in HEVC and its Extension to Compressed Sensing

July 2018

Waseda University
Graduate School of Fundamental Science and Engineering

Jianbin ZHOU
周　剣斌

Abstract

People have always wanted better visual experiences. From Ultra High Definition Television (UHDTV), 3D video to Virtual Reality (VR), the pursuit of overwhelming visual experiences is unlimited. For such visual experience, the higher resolution and frame rate are important. For instance, the 8K UHD (7680*4320 resolution) with 120 frames per second (fps) is considered for the video application of the next decade. It is reported the sports broadcasting and the perfect VR require even higher resolution and frame rate, which sets technical challenges, such as the huge data volume and high processing throughput (frame resolution * frame rate).

Such huge data are impossible to store or transfer without encoding. The video data to encode have two types, the pixels and measurements (linear combinations of pixels). Pixels are generated from traditional CMOS image sensor, and measurements from Compressed Sensing (CS) based CMOS image sensor. The pixel encoding has a long history since 1968. High Efficient Video Coding (HEVC) is the most advanced one achieving a high compression ratio at the expense of high computational complexity contributed by the new features. Hence, designing high performance VLSI architecture to support UHD video application are challenging and necessary. Among all components in HEVC, the VLSI architecture of Intra prediction and Sample Adaptive Offset (SAO) are chosen. Since they are the most different components in function comparing with H.264. The different requires new and efficient VLSI architecture to support the UHD video encoding. They are discussed and proposed in Chapter 2 and 3 respectively.

As the resolution and frame rate increase, the traditional image sensor has power consumption problem and higher frame rate is hard to achieve. These problems could be solved by new type of image sensor using CS. It could recover the whole image by capturing only few measurements in the image sensor. Capturing much less data instead of every pixel, the power consumption in the image sensor could be reduced, hence it provides a promising future for the increasing resolution and frame rate in video

application. Measurements coming from CS image sensor still required encoding before the transfer. However, measurements don't have the obvious spatial similarity that provides spaces for intra prediction in HEVC. To better encode the measurements, the intra prediction algorithm and VLSI architecture for CS is explored in Chapter 4 and 5.

VLSI architecture supporting high parallel degree (amount of pixels/measurements process per cycle) is necessary to processing the huge data. However, the higher parallel degree results into larger circuit area thus reducing the performance (Throughput / circuit area). This dissertation mainly targets on the high-performance VLSI architecture of HEVC SAO Estimation, intra prediction for encoder and its extension in Compressed Sensing, by using the proposed concept "reduced video data". Only by taking the necessary video data, including pixels and measurements, it is possible to reduce the parallel degree in hardware while keeping the performance during the data processing. The summary of each chapter is introduced as follows.

Chapter 1 [Introduction] introduces the big picture of video acquisition process, including the traditional imaging and the CS imaging. Next, HEVC intra prediction and SAO are introduced. Furthermore, the motivation to explore the intra prediction in CS is introduced. At last, proposed concepts of this dissertation are shown.

Chapter 2 [VLSI architecture of HEVC Intra prediction using reduced loaded-pixels] presents the high-performance VLSI architecture for HEVC intra prediction. Intra prediction uses neighboring pixels from different directions to predict pixels of a block (4x4~32x32). As the block size increases from 16 to 32 in HEVC, it takes 3x more neighboring pixels for prediction. Instead of loading all neighboring pixels as previous work, only the necessary pixels are loaded. This proposed idea reduces the two-third of reference pixels, thus reducing the area and increasing the throughput. It is achieved by LUT (Look Up Table) generated by software to tell which pixels are demanded in each prediction mode and location. Another proposal is the Hybrid Block Reordering and Data Forwarding, minimizing the idle time and eliminating the dependency between blocks by creating three Data Forwarding paths.

It achieves the hardware utilization of 94%. The proposed VLSI architecture has a gate count of 217.8K, and is able to support 4320p@120fps HEVC intra prediction.

Chapter 3 [Dual-clock VLSI architecture of HEVC Sample Adaptive Offset Estimation] presents a high-performance VLSI design for SAO estimation. SAO is a process to find out optimal offsets to reduce ringing noises in an image. It consists of two steps, Statistics Collection (SC) and Parameter Decision (PD), each of them has totally different nature in calculation. SC has huge but simple calculations while PD has few but complex calculations. After studying such nature, it is discovered that reducing pixels to process per clock cycle in SC significantly reduces the area. Thus, a dual-clock architecture is proposed, where SC works under high frequency and PD under low frequency, so that SC could process few pixels each cycle. Such proposal reduces the overall area by 56%. To further improve the area and power efficiency, algorithm-architecture co-optimizations are applied including a coarse range selection (CRS) and an accumulator bit width reduction (ABR). CRS shrinks the range of fine processed bands for the band offset estimation. ABR further reduces the area by narrowing the accumulators of SC. They together achieve another 25% area reduction. The proposed VLSI design is capable of processing 8K@120fps encoding. It occupies 51K logic gates, only one-third of the circuit area of the state-of-the-art design.

Chapter 4 [Algorithm and VLSI architecture of intra prediction in Compressed Sensing using reduced measurements] presents a measurement intra prediction framework. Instead of using all measurements for prediction, measurements for prediction are reduced to two. These two measurements embed the block boundary information of closest area. They are obtained by modifying two rows in the random 0/1 measurement matrix. Furthermore, a low-cost VLSI architecture is implemented for the proposed framework, by substituting the matrix multiplication with shared adders and shifters. The experimental results show that our proposed framework can compress the measurements and increase coding efficiency, with 34.9% BD-rate reduction compared to the direct output of CS-based sensors. The VLSI architecture of the proposed framework is 9.1K in area, and achieves the 83% reduction in size of memory bandwidth and storage for the line buffer. This could significantly reduce both the energy consumption and bandwidth in communication of wireless camera systems.

Chapter 5 [Row-Operation-Based Intra prediction under Approximate-DCT measurement matrices and its VLSI Architecture implementation] presents the row-operation to perform the intra prediction on the proposed approximate-DCT measurement matrices. Deterministic measurements matrices derived from approximated-DCT are proposed, significantly increasing the coding efficiency comparing with the random binary matrix in Chapter 4. However, the intra prediction using two measurements in the last chapter could not work on proposed matrices. Instead of using all measurements for prediction, the row-operation using three measurements are proposed. It achieves intra prediction as Chapter 4, without modifying the measurement matrix. Lastly, the VLSI architecture design for the intra prediction is proposed. Experiment results show the proposed matrix improve the coding efficiency by BD-PSNR increase of 4.2 dB. The proposed row operations increase the coding efficiency by 0.24 dB BD-PSNR. The VLSI architecture is only 4.3 K gates in area and 0.3 mW in power consumption, which is only half of the area and the power consumption in previous work.

Chapter 6 [Conclusions and future work] concludes the contributions of this dissertation. The solved and remaining problems are left for the future works.

Contents

Abstract .... **I**

Contents .... **V**

Index of Figures .... **VIII**

Index of Tables .... **X**

Acronyms .... **XI**

1. Introduction **1**

1.1 Ultra-High Definition video application and video coding. 1

1.2 HEVC encoding 4

1.3 Motivation on Intra prediction and SAO 5

1.4 Motivation of applying HEVC Intra prediction to Compressed Sensing 6

1.5 Proposed concept and target of this dissertation 8

2. VLSI architecture of HEVC Intra prediction using reduced loaded-pixels **13**

2.1 Introduction 13

2.2 Introduction to Intra Prediction in HEVC / H.265 16

2.2.1 Reference sample preparation 16

2.2.2 Intra Sample Prediction 17

2.3 Proposed VLSI Architecture 17

2.3.1 Overview 18

2.3.2 LUT-RSFG 20

2.3.3 4×4 Block Based Prediction 24

2.3.4 Hybrid Block Reordering and Data Forwarding 26

2.4 Implementation Results 29

2.5 Summary 32

3. Dual-clock VLSI architecture of HEVC Sample Adaptive Offset Estimation **33**

3.1 Introduction 33

3.2 SAO algorithm 35

3.2.1 Statistics Collection 36

3.2.2 Parameter Decision 38

3.2.3 Hardware friendly SAO 38

3.3 Dual-clock Architecture 40

3.3.1 Heterogeneous data flows of SC and PD 40

3.3.2 The optimal clock frequency of SC 41

3.3.3 Other feasible models for parallel SC 44

3.3.4 The optimal frequency in PD 46

3.3.5 Proposed architecture 47

3.4 Algorithm-Architecture Co-optimization 49

3.4.1 Coarse Range Selection 49

3.4.2 Accumulator bit width reduction (ABR) 51

3.5 Experimental Results 56

3.6 Summary 59

4. Algorithm and VLSI architecture of intra prediction in Compressed Sensing using reduced measurements 60

4.1 Introduction 60

4.2 Compressed Sensing (CS) 64

4.2.1 Concept of CS: 64

4.2.2 Process of CS image sensing: 65

4.3 Proposed Coding Framework Based on Measurement- Domain Intra Prediction 67

4.3.1 Proposed framework 67

4.3.2 Structural measurement matrix 68

4.3.3 Measurement-domain intra prediction 68

4.4 VLSI implementation of the proposed intra coding framework 70

4.5 Experimental results 72

4.6 Summary 83

5. Row-Operation-Based Intra prediction under Approximate-DCT measurement matrices and its VLSI Architecture implementation 85

5.1 Introduction 85

5.2 Approximate DCT and Compressed Sensing (CS) 88

5.2.1 Approximated DCT: 88

5.2.2 Concept of CS: 88

5.2.3 Process of CS image sensing and image reconstruction: 90

5.3 Proposed measurement matrix 91

5.3.1 The main idea 92

5.3.2 Derivation of proposed matrix Φd 93

5.3.3 The performance comparison 94

5.4 Proposed matrix row operation for measurement-based intra prediction and its VLSI architecture 96

5.4.1 Existing measurement-based intra prediction 96

5.4.2 Proposed matrix row operation for measurement-based intra prediction 98

5.4.3 Measurement-based intra prediction framework and its VLSI implementation ... 100

5.5 Experiment Results 102

5.6 Summary 105

6. Conclusions and future work 107

Acknowledgement 110

References 111

Publication List 120

Journal Paper (with review): 120

International Conference Paper (with review): 122

Index of Figures

Fig. 1 Traditional image acquisition procedure........1
Fig. 2 Compressed Sensing procedure........2
Fig. 3 Video data volume under different resolution and compression methods........3
Fig. 4 Comparison of data volume of two frameworks in CMOS Image Sensor and encoder. 4
Fig. 5 HEVC encoding........5
Fig. 6 The history and development of video coding tools existed in HEVC........6
Fig. 7 Motivation of applying HEVC intra prediction to Compressed Sensing........6
Fig. 8 Measurement encoding........7
Fig. 9 The big map of the dissertation is shown........9
Fig. 10 The architecture of the intra prediction........19
Fig. 11 An example to show how RSFS work. 20
Fig. 12 Analysis of the required number of reference samples for prediction of a 4×4 block 24
Fig. 13 The structure of one PE, designed for Planar, DC and angular predictions........24
Fig. 14 Dependency problem in blocks........26
Fig. 15 Three dependency cases in 4×4 TUs and 8×8 TUs........26
Fig. 16 Layout of the proposed intra architecture........30
Fig. 17 The overview of SAO........34
Fig. 18 EO patterns and categories. 36
Fig. 19 Thirty-two Bands division. 37
Fig. 20 An example to illustrate the statistics collection process........37
Fig. 21 Samples not used in statistics collection, referring to No.6 in Table 7........40
Fig. 22 The architecture of 2×2 statistics collection engine........42
Fig. 23 The proposed architecture of parameter decision Engine........45
Fig. 24 The relationship among parallelism (N), area and frequency in SC. 46
Fig. 25 The schedule of CTB processed in pipeline in two clock domains. 48
Fig. 26 The number of samples distributed in each band within a CTB. 48
Fig. 27 The distribution of number of bands in a CTB in various sequences. 49
Fig. 28 The process of reduced BO candidates searching........51
Fig. 29 Layout of the proposed SAO architecture. 58
Fig. 30 Data flow of imaging........63
Fig. 31 Brief architecture of CS-CIS........63
Fig. 32 Process of CS-based image sensing, encoding, and decoding........65
Fig. 33 Predictor candidates........66
Fig. 34 Proposed artificially structured rows. 66
Fig. 35 VLSI architecture for the proposed measurement-domain intra prediction framework. 70
Fig. 36 Example of random binary matrix of N = 4........70
Fig. 37 Comparison of the number of measurements stored in the memory for intra prediction. 72
Fig. 38 Comparison of MSE of residuals in each block among three algorithms........75

Fig. 39 BD-rate curve of three test images.......81
Fig. 40 Visual comparison among Prop., SDPC, and CP (left, middle, right). 82
Fig. 41 A brief architecture of a processing block in CS-CIS [71] (Component A in Fig. 42). Outputs are digital measurements Y. 85
Fig. 42 The process of CS based image sensing, encoding and decoding. 90
Fig. 43 (a) An image is separated block by block. (b) A 2D-DCT transform, N=4. The blue dashed line shows the Z-scan order, ascending frequency response. (c) The 1D representation of (b). (d) An example of N = 4, m = 4 measurements are taken in the Z-scan order.. 91
Fig. 44 The BD curve comparison between proposed measurement matrix with Z-scan and N-scan, using two images, mandrill (left) and F16 (right). Sample rate (SR) of 0.25 (first row) and 0.50 (second row) are evaluated. Matrices are generated the approximate DCT from several previous work (N=4: [68], 8: [69] and 16: [70]). The marker Circle, Plus and Square represent N=4,8 and 16 respectively. Blue dash line represents N-scan and green solid line Z-scan 95
Fig. 45 The predictor candidates (Blue: Bottom row of upper block and 96
Fig. 46 A 4×4 block as input signal X. The target is to extract the sum of the blue and red part. (b) In the existing method [60][66], the first two rows in the random binary matrix are modified as rBR and rRC to extract the information (sum of bottom row and sum of rightmost column) 97
Fig. 47 The proposed matrix row operation performed on the proposed matrix Φt to generate rBR and rRC to extract the sum of bottom row and the sum of rightmost column 98
Fig. 48 VLSI architecture (Component B and C in Fig. 42) for the measurement-based intra prediction with proposed matrix row operation. The dashed boxes are marked by numbers in parentheses, corresponding to the equations (49)–(54) in Section 4. The matrix multiplication in the gray solid box could be simplified into the shift operation. 101
Fig. 49 The BD curve of lena, barbara, mandrill and F16 (top to bottom) with N=4 and N=8, SR =0.5, with various Q step $\in$ [0, 6]. The proposed measurement matrix in Prop. and Prop.+ are trimmed by Z-scan. 102
Fig. 50 The visual quality comparison. Left :Prop.+ Right: Dir. 104
Fig. 51 Comparison of MSE of residuals in each block among three algorithms, with N = 4 and 8, SR = 0.5, and Q$step$ =4 of lena (first row) and mandrill (second row) 104

Index of Tables

Table 1 Comparison of compression ratio..3
Table 2 The selection of reference samples and SRAM banks, given (X,Y)=(0,3), and 16×16 TU (Bold text represents the samples for filtering)...18
Table 3 SRAM Deployment for top, left and top-left luma samples......................................23
Table 4 Worst throughput cases and processing speeds 29
Table 5 Comparison of H.265/HEVC intra prediction architectures for video application 31
Table 6 Output (parameter sets) of SAO. 36
Table 7 Comparison of modifications made on the HM-16.0 39
Table 8 Relationship between the consumption of hardware resource of crucial modules and the increase of N from 1 to 32. (∗: when applying proposals in section 3.4, this figure is 48, else it is 96.) 44
Table 9 The BD rate comparison of algorithms under LowDelay_ain_P (LDP) configuration (Anchor : HM-16.0, SAO off, CTB:64×64)...53
Table 10 The BD rate comparison of proposed algorithm with CRS and ABR under different configuration, All Intra(AI),Random Access(RA), and LowDelay main(LD) (Anchor : HM-16.0, SAO on, CTB :64×64)..54
Table 11 Comparison of synthesis result with the previous works 55
Table 12 Comparison of the proposals with and without CRS and ABR................................55
Table 13 (a) BD-PSNR (BD-Rate) comparison. (Anchor is CP, with $Qstep \in [0,6]$ and N=4. Reconstruction algorithm is L1-PD with DCT) 75
Table 14 (b) BD-PSNR (BD-Rate) comparison. (Anchor is CP, with $Qstep \in$ [0,6] and N=8. Reconstruction algorithm is L1-PD with DCT) 77
Table 15 (c) BD-PSNR (BD-Rate) comparison. (Anchor is CP, with $Qstep \in$ [0,6] and N= 16. Reconstruction algorithm is L1-PD with DCT) 78
Table 16 Performance of the architecture. 78
Table 17 BD-PSNR Comparison. 105
Table 18 Performance of VLSI architecture...105

Acronyms

ABR: Accumulator bit width reduction
ADC: Analogy-to-Digital Conversion
AI: All intra
AVC: Advanced Video Coding
BD-Rate: Bjøntegaard bit-rate
BD-PSNR: Bjøntegaard peak signal to noise ratio
BO: Band Offset
BR: bottom row
CABAC: context-adaptive binary arithmetic coding
CIS: CMOS Image Sensor
CP: constant prediction
CRS: Coarse range selection
CS: Compressed Sensing
CS-CIS: Compressed Sensing based CMOS Image Sensor
CTB: Coding Tree Block
CTU: Coding Tree Unit
CU: Coding Unit
DC: Direct current
DCT: discrete cosine transform
DVC: Distributed video coding
DWT: discrete wavelet transform
EO: Edge Offset
fps: frame per second
HBRDF: Hybrid Block Re-ordering and Data Forwarding
HD: high definition
HEVC: high efficiency video coding
IoT: Internet of Things
ILF: in-loop filters
JCT-VC: Joint Collaborative Team on Video Coding
JPEG: joint picture experts group
L1-PD: L1 norm primal–dual interior–point method
LD: Low Delay
LUT: look-up table
LUT-RSFS: look-up table based Reference Sample Fetching Scheme
LUT-RSPG: LUT based Reference Sample Position Generator
M2M: Machine-to-Machine
MPEG: moving picture experts group
PD: parameter decision
PE: processing element
PLL: phase-locked loops
PSNR: peak signal to noise ratio

Acronyms

PU: prediction unit
RA: Random Access
RC: right-most column
RD: rate-distortion
RDO: rate distortion optimization
SAD: sum of absolute difference
SAO: Sample Adaptive Offset
SC: Statistics Collection
SDPC: spatially directional predictive coding
SoC: System on Chips
SR: sample rate
SRAM: Static Random Access Memory
TU: Transform Unit
TB: Transform Block
UHDTV: Ultra High Definition Television
VLSI: Very large scale integrated
VR: Virtual Reality

1. Introduction

1.1 Ultra-High Definition video application and video coding.

People have always wanted better visual experiences. From Ultra High Definition Television (UHDTV), 3D video to Virtual Reality (VR), the pursuit of overwhelming visual experiences is unlimited. For such visual experience, the higher resolution and frame rate are important. For instance, the 8K UHD (7680*4320 resolution) with 120 frames per second (fps) is considered for the video application of the next decade. It is reported the sports broadcasting and the perfect VR require even higher resolution and frame rate, which sets technical challenges, such as the huge data volume and high processing throughput (frame resolution * frame rate).

Such huge data are impossible to store or transfer without encoding. Video contents to encode have two types, the pixel and the measurement (linear combinations of pixels). Pixels are generated from traditional CMOS image sensor (CIS), and measurements from Compressed Sensing (CS) based CMOS image sensor. For the pixel, CIS senses the analog pixels one by one and converts them into digital ones; the encoder encodes pixels into much smaller bit stream, by encoding methods such as JPEG [47], H.264/MPEG- 4 [48], DVC [49], and HEVC / H.265 [7] and transmits the bit stream to the decoder. This is the traditional procedure: Capture → Compress → Transmit, as shown in Fig. 1.

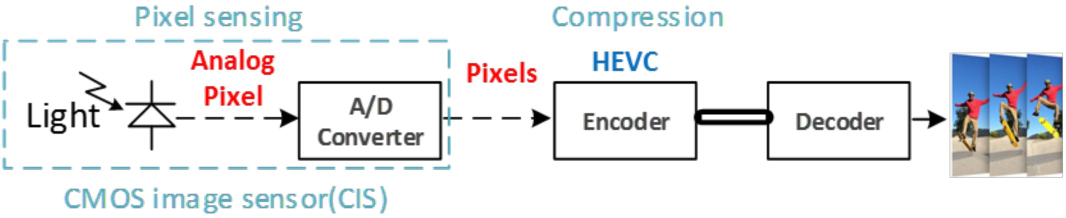


Fig. 1 Traditional image acquisition procedure

With the advent of a recently proposed sampling theory, Compressed Sensing (CS)

[51], the capturing and compression can be performed in CIS simultaneously. Such image sensors are called CS-based CIS (CS-CIS). In CS-CIS, an image is acquired by sampling a significantly reduced number of measurements (the linear combination of pixels), instead of sampling every pixel. The encoder takes measurements as input and compress them into smaller bit stream, and transmit to the decoder, as shown in Fig. 2. This technique could reduce the throughput of Analog-to-Digital (A/D) conversion, since the number of conversion is reduced by sampling the measurement instead of pixel. The reduction in throughput has the potential to reduce power consumption and increase the frame rate [52], which has been shown in the recently emerging CS-CIS systems [53][54][55].

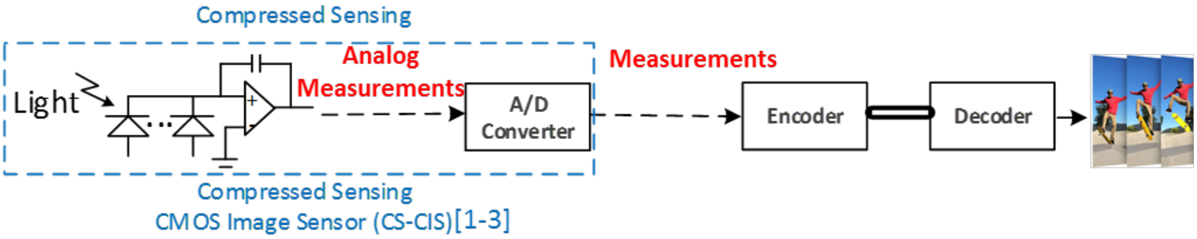


Fig. 2 Compressed Sensing procedure

For traditional image acquisition, its merit is that pixel encoding could achieve a high compression ratio, as HEVC, the latest, the most advanced and the most complex video coding standard, achieving the compression ratio up to 1/200 as shown in Table 1, which is 50% coding efficiency increase comparing with the previous video coding standard, H.264. Compared with storing and transferring the raw video data, it could solve the huge data amount issues in storage and transferring. Furthermore, another merit is that the compression method is easier to design, since the spatial and temporal similarity could be obviously exploited. The demerit is that, however, it has high computation complexity and gives challenges in VLSI implementation, resulting into difficulties to achieve high throughput and low hardware-cost. Another demerit is that, the throughput and the power consumption of ADC dominates the power consumption in the CIS with the increase of the resolution and frame rate [54][53] [55].

Table 1 Comparison of compression ratio.

Coding standard	History (since)	Compression ratio
MPEG-2	1995	1/50
H.264	2003	1/100
HEVC	2013	1/200

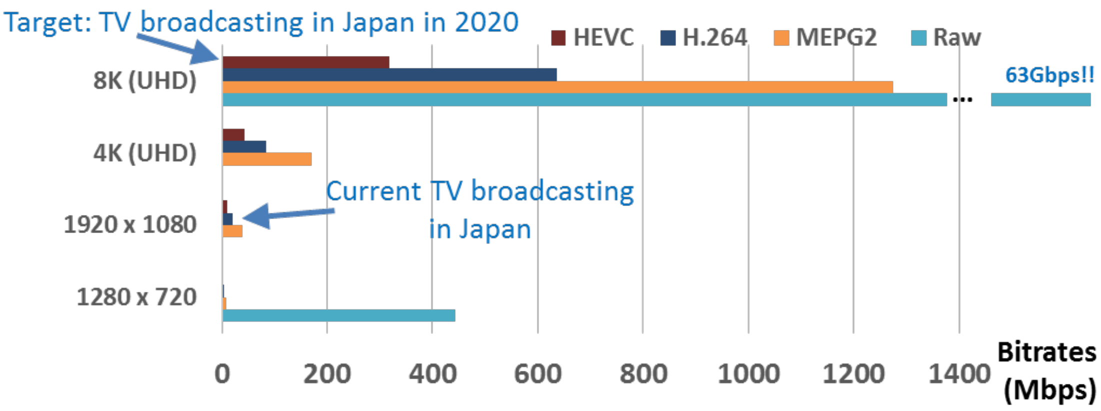


Fig. 3 Video data volume under different resolution and compression methods

For CS acquisition, the merit is obvious because it could greatly reduce the throughput and its power consumption of image acquisition in camera, which is very potentially suitable for the increasing throughput in mobile video application. The reduced throughput of ADC could reduce the power consumption of CIS and increase the possibility to achieve higher frame rate. Since the output of CS-CIS is measurement instead of pixel. However, measurements generated in CS is not possible to encode by the traditional video coding, which is one of the demerits. Hence, the current existing encoding methods are developing and do not have an ideal coding efficiency.

The comparison of two image acquisition framework is shown in Fig. 4. On one hand, the traditional image acquisition framework with traditional video encoding, HEVC, achieves a significant compression ratio, at the expense of extremely high computational complexity. On the other hand, the CS framework could reduce the data volume of A/D conversion, but its output is not applicable to traditional video coding. Hence it has lots of spaces to develop new and better video encoding methods applicable to CS.

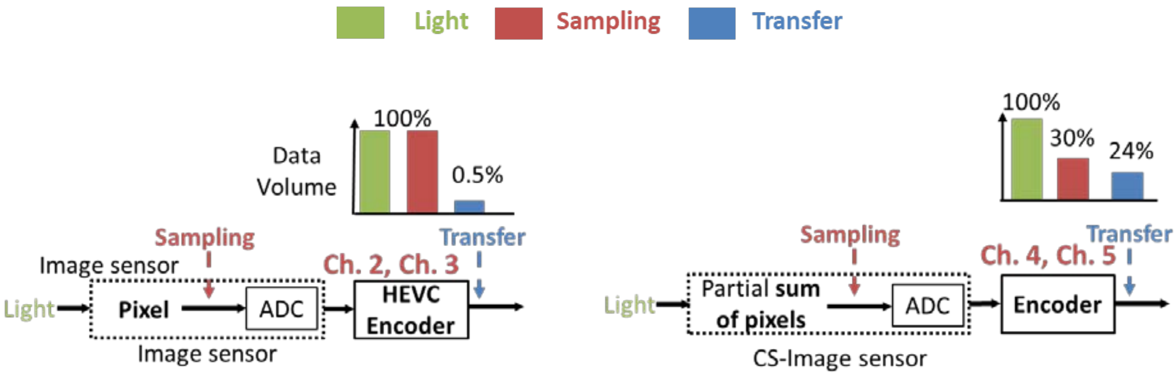


Fig. 4 Comparison of data volume of two frameworks in CMOS Image Sensor and encoder.

1.2 HEVC encoding

To transmit such a huge data throughput in the communication channel, deep compression from the latest video coding technology, High Efficiency Video Coding (H.265/HEVC) [7][20], plays a crucial role. The implementation of the corresponding video codecs, however, is challenged by the multiplication of the ultra-high definition requirement and an increased complexity per pixel. Compared to the previous H.264/AVC standard [21], H.265/HEVC doubles coding efficiency by employing a number of new coding tools.

The encoding of HEVC is to compress sequences of images into the bit stream. It includes the following components. First, the transform and quantization, as shown in the yellow component in Fig. 5. It has been proposed in [1] and [2] that the image energy compaction exists through transform, such as Discrete Cosine Transform (DCT) or Hadamard Transform, such that most of the energy are concentrated in the low frequency components. Since human’s eyes are more perceptive to the low frequency components and not sensitive to the high frequency components, the image data are compressed by preserving the low frequency components with a higher accuracy while the high frequency components with a lower accuracy.

Second, it was further proposed in [3] that the data could be further compressed by exploiting the spatial and temporal data redundancy, such as the inter prediction and intra prediction shown in Fig. 5. Hence, the residuals instead of the original pixels are taken as the input of the transform and quantization. Because of data loss introduced by the quantization, the reconstructed pixels in the decoder get different from the original pixels in the encoder. To guarantee a successful reconstruction, the same reconstruction path as the one in the decoder is built in the encoder.

Third, image artifacts are generated by the quantization. The De-Blocking Filter (DBF) proposed in [4] and [5] are used for removing the blocking artifacts existing on the boundary of blocks. The Sample Adaptive Offset (SAO) Estimation, and SAO Filter proposed in [22] is used for removing the artifacts existing on the edge region of an image. At last, all the data and signals are coded into bit stream by an entropy coder, Context-based Adaptive Binary Arithmetic Coder (CABAC).

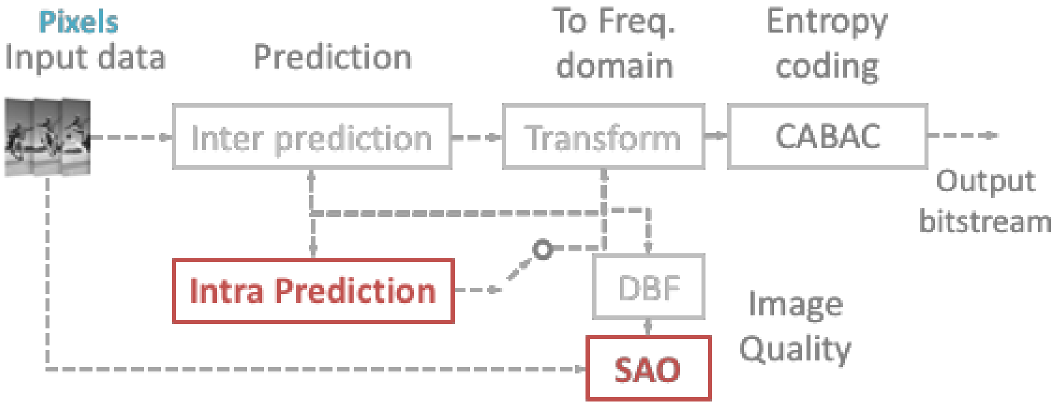


Fig. 5 HEVC encoding

1.3 Motivation on Intra prediction and SAO

The components introduced above are mostly inherited from the previous video coding standard. Their history and development are shown in Fig. 6. The VLSI architecture design for these components has been proposed and improved in the past, and many components are well designed in the previous standard. The coding efficiency of HEVC is improved by new features being added in each component, at the expense of higher computational complexity. Some components have big changes while some have not. Among all the components, Intra and SAO consist of the major changes. Intra prediction consists of most of the new features comparing with H.264. SAO is a new coding tool that does not exist in the previous standard, H.264. Because of the new functions, the increase of computational complexity in intra prediction and SAO as well as their huge difference from the previous standard, there're many challenges for designing the high-performance VLSI architecture supporting the UHD video application.

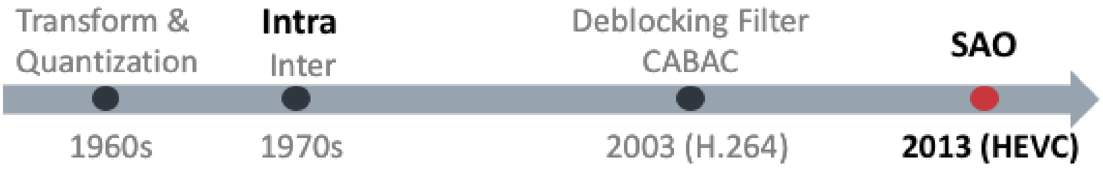


Fig. 6 The history and development of video coding tools existed in HEVC

Intra prediction contributes to 22%-36% bitrate saving comparing with H.264 by introducing new changes at the expense of the increased computational complexity, which is reflected in three aspects. First, more block sizes are used in intra prediction of H.265. Second, more intra prediction modes in each size of block are utilized. Finally, more filtering methods that depend on the prediction modes and sizes of prediction blocks are used. These changes increase the difficulty to achieve the high-performance architecture.

1.4 Motivation of applying HEVC Intra prediction to Compressed Sensing

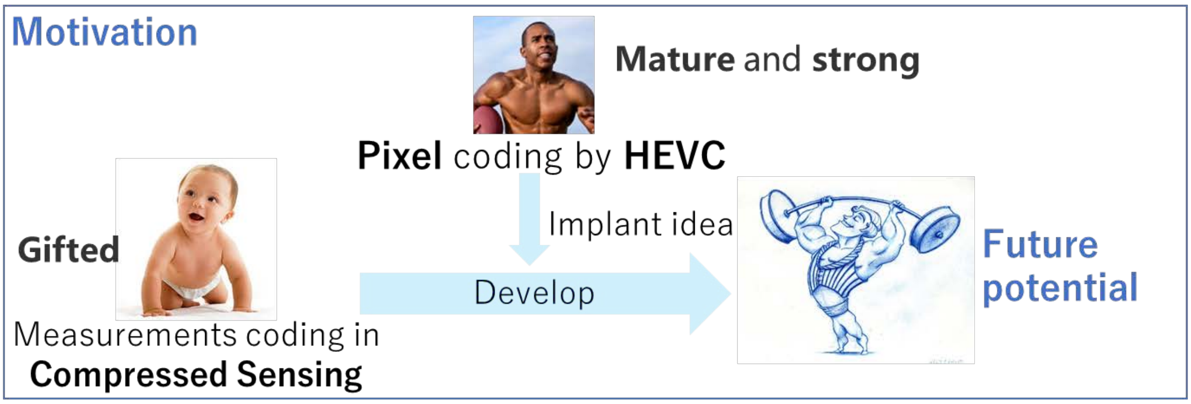


Fig. 7 Motivation of applying HEVC intra prediction to Compressed Sensing

As the resolution and frame rate increase, the traditional image sensor has a power consumption problem and higher frame rate is hard to be achieved. Even the HEVC has high coding efficiency, it could not reduce the power and frame rate problem in image sensor. Because image sensor is necessary for video to be displayed, such problems would influence the further development of video application with higher resolution and higher frame-rate. With the advent of CS theory, these problems could be solved by a new type of image sensor using CS. It could recover the whole image by capturing only few measurements in the image sensor. Capturing much less data instead of every pixel, the power consumption in the image sensor could be reduced, hence it provides a promising future for the increasing resolution and frame rate in video application. However, HEVC does not work on measurements in CS. To make the CS image sensor could be widely used in UHD video application, effective encoding algorithm and VLSI architecture applicable to CS is necessary.

The encoding framework of CS is shown in Fig. 8. Since the input of the encoder is measurements generated by the CS-CIS, the encoding procedure begins from taking measurements block by block as input. Instead of transferring the original measurements, the optimal residuals are chosen and transferred to the quantization. Due to the quantization error, the reconstructed measurements are not the same as the original one. To guarantee the functional consistency in decoder and encoder, the reconstruction loop also exists, as shown in the light green part. The framework of measurement encoding is similar to the one in Fig. 5, and it is also much simpler.

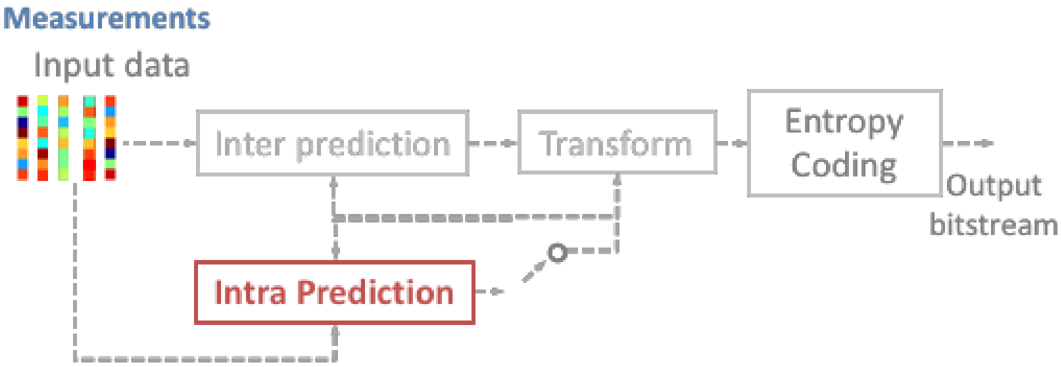


Fig. 8 Measurement encoding

1.5 Proposed concept and target of this dissertation

VLSI architecture supporting high parallel degree is necessary to real-time processing the UHD video data. This dissertation mainly targets on the high-performance VLSI architecture of HEVC SAO Estimation, intra prediction and its exploration in Compressed Sensing. Increasing the parallel degree in VLSI could meet the high throughput requirement, while sacrificing the hardware cost.

The concept "reduced video data" is proposed. Only by taking the necessary video data, including pixels and measurements, it becomes possible to reduce the parallel degree in hardware while keeping the performance during the data processing. The VLSI architecture of HEVC Intra prediction and Sample Adaptive Offset (SAO) Estimation for 8K@120fps video encoding are discussed and proposed in Chapters 2 and 3. In intra prediction, the "reduced-loaded-pixels" is proposed. Only the necessary pixels for prediction is loaded from memory, instead of all pixels to load as previous work. In SAO estimation, the optimal clock frequency is discovered, so that the optimal number of pixels processed per cycle in Statistics Collection could be achieved. The dual-clock VLSI architecture work on Statistics Collection and Parameter Decision separately to make the calculation in both stage efficient, so that the hardware cost for calculation is reduced. The exploration of HEVC intra prediction to CS is discussed in Chapters 4 and 5. The proposed algorithm find-out the possibility to reduce the number of measurements for intra prediction. The number of measurements is reduced to the constant number from the scale that is quadratic growth with the block size as previous work. They only consist of the local information of a block for prediction, instead of global information for prediction as previous work. It also includes proposed algorithms to improve the coding efficiency by the proposed measurement matrices and row-operation on the matrix. The low-cost VLSI supporting 4K@240fps UHD video encoding is also proposed. The big map of the dissertation is shown figure below.

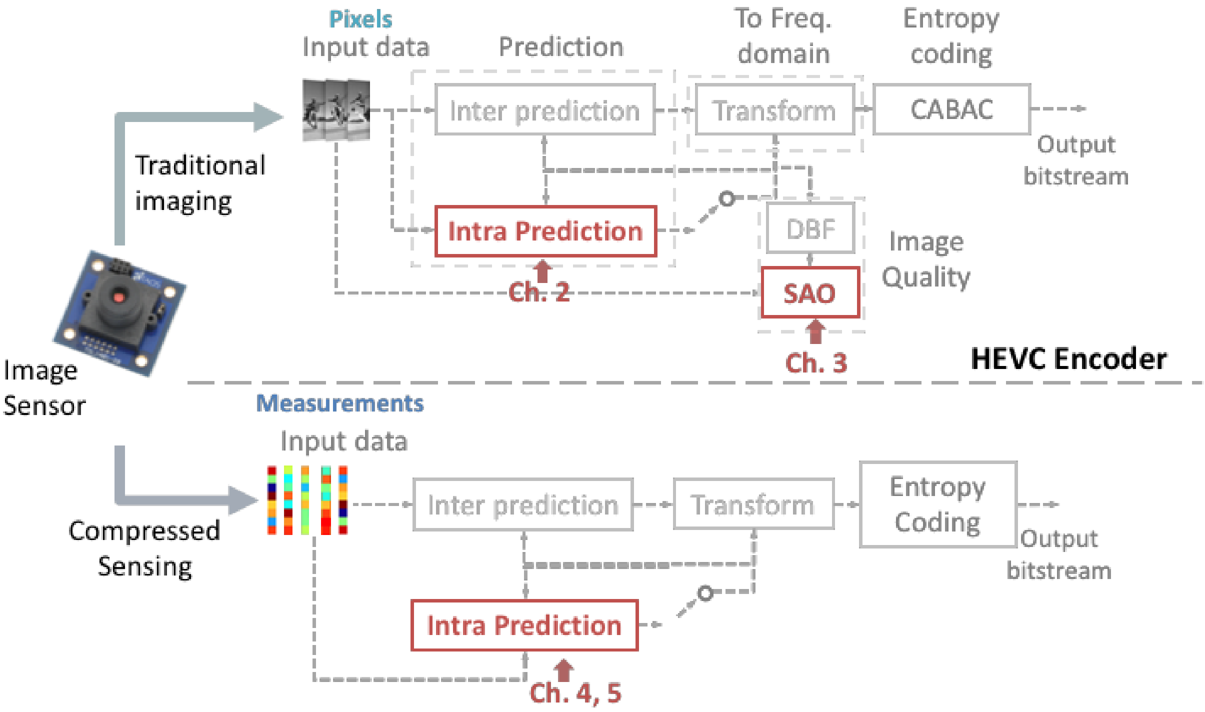


Fig. 9 The big map of the dissertation is shown.

The summary of each chapter is introduced as follows.

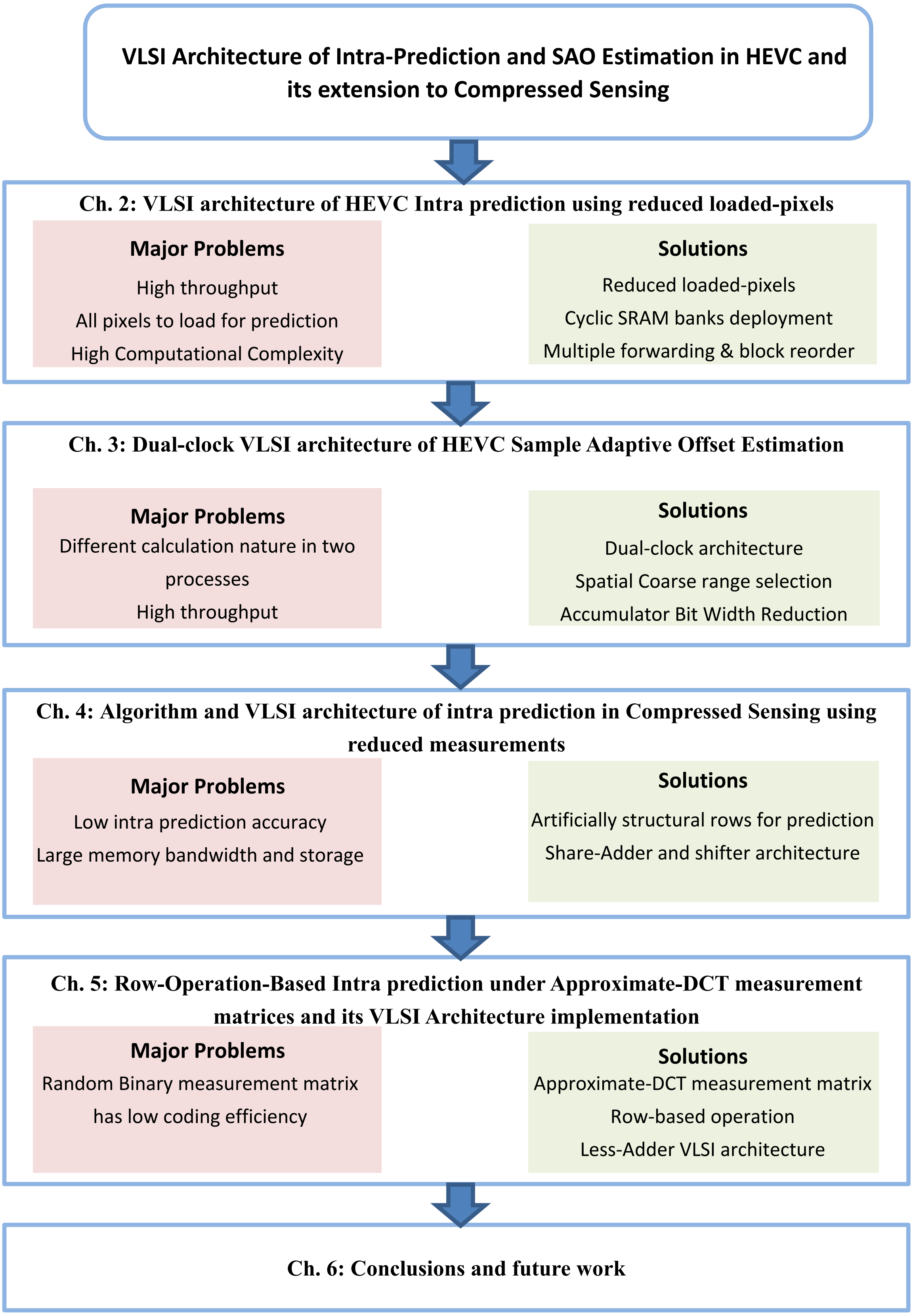
Chapter 2 [VLSI architecture of HEVC Intra prediction using reduced loaded-pixels] presents the high-performance VLSI architecture for HEVC intra prediction. Intra prediction uses neighboring pixels from different directions to predict pixels of a block (4x4~32x32). As the block size increases from 16 to 32 in HEVC, it takes 3x more neighboring pixels for prediction. Instead of loading all neighboring pixels as previous work, only the necessary pixels are loaded. This proposed idea reduces the two-third of reference pixels, thus reducing the area and increasing the throughput. It is achieved by LUT generated by software to tell which pixels are demanded in each prediction mode and location. Another proposal is the Hybrid Block Reordering and Data Forwarding, minimizing the idle time and eliminating the dependency between blocks by creating three Data Forwarding paths. It achieves the hardware utilization of 94%. The proposed VLSI architecture has a gate count of 217.8K, and is able to support 4320p@120fps HEVC intra prediction.

Chapter 3 [Dual-clock VLSI architecture of HEVC Sample Adaptive Offset Estimation] presents a high-performance VLSI design for SAO estimation. Its consists of two processes, statistics collection (SC) and parameter decision (PD), each of which demands difference frequency. After investigating the optimal frequency, a dual-clock architecture is proposed to deal with SC and PD with different speed of clocks. Such a strategy reduces the overall area by 56%. To further improve the area and power efficiency, algorithm-architecture co-optimizations are applied including a coarse range selection (CRS) and an accumulator bit width reduction (ABR). CRS shrinks the range of fine processed bands for the band offset estimation. ABR further reduces the area by narrowing the accumulators of SC. They together achieve another 25% area reduction. The proposed VLSI design is capable of processing 8K@120fps encoding. It occupies 51K logic gates, only one-third of the circuit area of the state-of-the-art design.

Chapter 4 [Algorithm and VLSI architecture of intra prediction in Compressed Sensing using reduced measurements] presents a measurement-domain intra prediction framework. Instead of using all measurements for prediction, measurements for prediction are reduced to two. These two measurements embed the block boundary information of closest area. They are obtained by modifying two rows in the random 0/1 measurement matrix. Furthermore, a low-cost VLSI architecture is implemented for the proposed framework, by substituting the matrix multiplication with shared adders and shifters. The experimental results show that our proposed framework can compress the measurements and increase coding efficiency, with 34.9% BD-rate reduction compared to the direct output of CS-based sensors. The VLSI architecture of the proposed framework is 9.1K in area, and achieves the 83% reduction in size of memory bandwidth and storage for the line buffer. This could significantly reduce both the energy consumption and bandwidth in communication of wireless camera systems.

Chapter 5 [Row-Operation-Based Intra prediction under Approximate-DCT measurement matrices and its VLSI Architecture implementation] presents the row-operation to perform the intra prediction on the proposed approximate-DCT measurement matrices. Deterministic measurements matrices derived from approximated-DCT are proposed, significantly increasing the coding efficiency comparing with the random binary matrix in Chapter 4. However, the intra prediction using two measurements in the last chapter could not work on proposed matrices. Instead of using all measurements for prediction, the row-operation using three measurements are proposed. It achieves intra prediction as Chapter 4, without modifying the measurement matrix. Lastly, the VLSI architecture design for the intra prediction is proposed. Experiment results show the proposed matrix improve the coding efficiency by BD-PSNR increase of 4.2 dB. The proposed row operations increase the coding efficiency by 0.24 dB BD-PSNR. The VLSI architecture is only 4.3 K gates in area and 0.3 mW in power consumption, which is only half of the area and the power consumption in previous work.

Chapter 6 [Conclusions and future work**]** concludes the contributions of this dissertation. The solved and remaining problems are left for the future works.



VLSI Architecture of Intra-Prediction and SAO Estimation in HEVC and its extension to Compressed Sensing

Ch. 2: VLSI architecture of HEVC Intra prediction using reduced loaded-pixels

Major Problems

High throughput

All pixels to load for prediction

High Computational Complexity

Solutions

Reduced loaded-pixels

Cyclic SRAM banks deployment

Multiple forwarding & block reorder

Ch. 3: Dual-clock VLSI architecture of HEVC Sample Adaptive Offset Estimation

Major Problems

Different calculation nature in two processes

High throughput

Solutions

Dual-clock architecture

Spatial Coarse range selection

Accumulator Bit Width Reduction

Ch. 4: Algorithm and VLSI architecture of intra prediction in Compressed Sensing using reduced measurements

Major Problems

Low intra prediction accuracy

Large memory bandwidth and storage

Solutions

Artificially structural rows for prediction

Share-Adder and shifter architecture

Ch. 5: Row-Operation-Based Intra prediction under Approximate-DCT measurement matrices and its VLSI Architecture implementation

Major Problems

Random Binary measurement matrix has low coding efficiency

Solutions

Approximate-DCT measurement matrix

Row-based operation

Less-Adder VLSI architecture

Ch. 6: Conclusions and future work

2. VLSI architecture of HEVC Intra prediction using reduced loaded-pixels

2.1 Introduction

H.265/High Efficiency Video Coding (HEVC) [1]-[7] is the most recent video coding standard, developed by Joint Collaborative Team on Video Coding (JCT-VC).With the same video quality, 40–50% bit rate reduction is achieved com- pared with H.264/AVC (Advanced Video Coding) standard [8]. Intra prediction plays an important role in H.265, saving about 22–36% of the bitrate. On one hand, intra prediction in H.265 is still based on blocks, and uses neighboring samples' values to calculate the values of the new blocks, which is similar to H.264. On the other hand, new changes are introduced in intra prediction of H.265. These new features help achieve higher coding efficiency at the expense of an increased computational complexity, which is reflected in three aspects. First, more block sizes are used in intra prediction of H.265. Second, more intra prediction modes in each size of block are utilized. Finally, more filtering methods that depend on the prediction modes and sizes of prediction blocks are used.

To support the real-time application of a higher resolution video, the system needs to process data faster. To achieve real-time application of 8K UHD video in intra prediction, the system needs to support 16x throughput comparing with HD video processing, and 4x comparing with 4K UHD video processing. Overall, more data have to be processed within a certain time.

Several hardware designs have been proposed for H.265/HEVC intra prediction. Li in [10] exploited an efficient uniform architecture for 4×4 blocks. This work is the first VLSI design try in intra prediction of HEVC. Huang in [11] proposed a memory-hierarchical and mode-adaptive architecture for 4K Ultra HD HEVC, which has a low circuit area. Jung [12] proposed an architecture for intra samples prediction; however, this architecture does not include key functions such as reconstruction or substitution. In [13], Palomino proposed an architecture that employs less-multiplier pipelines to increase the throughput and support all Prediction Units (PU) sizes. The architecture proposed by Liu in [14] applied a post-order traversal to the quad-tree structure targeting to reduce the internal buffers in the encoder. It supports all modes and all PU sizes for the 1080p@30fps HEVC encoder. In [15], Zhou reclassified the prediction modes to reduce the number of reference registers for full HD encoding. Among earlier implementations for H.264 intra prediction, He in [16] proposed the MB/block level co-reordering scheme to avoid data dependency. Amongst cited works, [11] and [12] investigate the decoder's design, while [13][14][15] and [16] investigate the encoder's design. These works did not solve the 8K video application issue.

The design of the architecture of the intra prediction engine depends on how the reference samples for prediction are fetched and how they are processed. A conventional way to deal with this problem in H.264 or HEVC is to fetch all the reference samples for processing a Transform Unit (TU) and store them in registers in advance. This method has been employed in many previous studies, [11][13][14][15]. The advantage of this method is the reduction of the number of accesses to the external memory system (on-chip/off- chip memory). However, it has two drawbacks. First and most important, prediction is done by selecting reference samples among these registers, and with the increase of registers, multiplexing would be more complex; thus, further increasing the critical path delay and circuit area. Since the critical path ought to be short enough to support a high performance for 8K UHD, this drawback would be a critical problem for 8K application. Furthermore, many registers to store all the reference samples in the preparation stage is required. For instance, for a 32×32 TU, as there are 32 samples in each for the left, top, and top-right neighboring samples, a minimum of number of 99 eight-bit registers are required. The large number of registers would increase the area of circuit. These two drawbacks make us design our architecture based on a different strategy from previous work. The details are shown in latter part.

To design intra prediction architecture for the 8K UHD H.265 application, we face two key challenges that did not exist in previous work [11]. The first challenge is the computational complexity in H.265 intra prediction shown in two aspects. First, a large number of reference pixels (up to 99 pixels) have to be loaded for prediction, which is four times more than that in H.264. The number of registers for storage increases the size and complexity of the circuit, causing performance reduction. Moreover, many modes and filtering methods have to be supported. The latter places more restrictions on loading reference pixels, making the system harder to design and implement.

The second challenge is that 8K@120fps UHD real time application requires an architecture that allows a high throughput. However, in intra prediction, the dependency between the processed and unprocessed TU obstructs the system from achieving a high throughput.

The proposed architecture is based on our main idea, divide-and-conquer strategy. The system does not have to fetch all reference samples before prediction, because all of them cannot be used immediately. By fetching a small required part for prediction first, and keeping fetching the others successively, we could enhance the performance and reduce the circuit area. This paper expands on our previous work in [17], and shows a more complete design. Main contributions of this paper are outlined as follows:

- I proposed the first technique, look-up table (LUT) based Reference Sample Fetching Scheme (LUT-RSFS), based on the divide-and-conquer strategy. It reduces the number of fetched reference samples in worst case from 99 to 13, such that the performance is improved and the circuit area is reduced.
- I proposed the second technique, Hybrid Block Re-ordering and Data Forwarding (HBRDF). The 4x4 block level reordering solves the dependency problem among 16×16, 32×32 TU. Three paths are created for Data Forwarding, to eliminate the dependency of 4×4 and 8×8 TU. Finally, a high throughput and hardware utilization of 94% are achieved.

The rest of the paper is organized as follows. First, Section 2.2 introduces the new features for H.265 intra prediction. Then, Section 2.3 describes the data flow of our proposal, and its details. Section 2.4 discusses the implementation results and finally

Section 2.5 concludes the paper.

2.2 Introduction to Intra Prediction in HEVC / H.265

Intra prediction is an important part in the video coding standard. It refers to the neighboring samples of previous coded blocks to reduce spatial redundancy. There are three types of blocks, Coding Unit (CU), PU and TU in H.265. When a frame is coded, it is divided into CUs and each root CU can be recursively divided into or four smaller CUs. Each leaf CU will be processed by PUs and TUs. PU ranges from 4×4 to 64×64, while TU from 4×4 to 32×32. If a CU is encoded in intra mode, each TU corresponds to an intra prediction block with the corresponding PU’s prediction mode. Therefore, the block’s size is from 4×4 to 32×32, and 35 prediction modes exist for intra prediction. Moreover, various reference sample filtering and substitution methods are added to intra prediction in H.265, which do not exist in the case of H.264/AVC.

2.2.1 Reference sample preparation

Before predicting a TU, a part of the neighboring samples (left, left-bottom, top-left, top, and top-right regions) are loaded. These regions may be located out of the frame, in other slices or tiles, or in the blocks that are not yet reconstructed. In such cases, they are marked as unavailable for intra prediction. In addition, when the constrained intra prediction is enabled, the neighboring inter-predicted blocks are also marked as unavailable for intra prediction. At that time, the nearest available neighboring sample from the unavailable ones is used as a substitute.

When the TU's size is larger than 4×4, its neighboring reference samples are filtered before being used for prediction. There are two types of filtering in H.265 intra prediction: three-tap finite impulse response (FIR) filtering and bilinear filtering. The bilinear filtering is used in 32×32 TU, when discontinuity is detected.

2.2.2 Intra Sample Prediction

I classify thirty-five intra prediction modes into three classes: angular modes, DC mode, and Planar mode. Each of them has a formula, which are stated below.

- For the angular modes, each predicted sample is calculated according to the equation:

  $pred. = ((32 - w) * ref\,[a] + w * ref\,[a + 1] + 16) >> 5$ (1)

  For angular modes, 2 to 10, and 26 to 34, consecutive neighboring reference samples are used for prediction. However, for angular modes 11 to 25, some discontinuous reference samples are used in the extended part.

- For the DC mode, the average of the top and left reference samples' value (dcVal) is used as the value for the whole PU.

  To remove discontinuities along block boundaries, the boundary samples are filtered in the DC mode, and in the angular mode 10 (horizontal) and 26 (vertical) when the luma Transform Block (TB) size is less than 32. The samples in the first column, first row and the top-left pixel are replaced by a two-tap FIR filter, fed by their adjacent reference sample and their original value.

- For the Planar mode, an order-2 plane prediction mode of H.264 is used. It is defined as the average of two linear predictions, as shown in (4), where N is the size of the TU, while x,y=0, . . . , N-1.

$$\mathrm{P}^{\mathrm{V}}_{x,y} = (\mathrm{N} - \mathrm{y}) * \mathrm{R}_{x,0} + \mathrm{y} * \mathrm{R}_{0,N+1} \tag{2}$$

$$\mathrm{P}^{\mathrm{H}}_{x,y} = (\mathrm{N} - \mathrm{x}) * \mathrm{R}_{0,y} + \mathrm{x} * \mathrm{R}_{\mathrm{N+1,0}} \tag{3}$$

$$\mathrm{P}_{x,y} = (\mathrm{P}^{\mathrm{V}}_{x,y} + \mathrm{P}^{\mathrm{H}}_{x,y} + \mathrm{N}) >> (\log 2\,(\mathrm{N}) + 1) \tag{4}$$

2.3 Proposed VLSI Architecture

This section is divided into four subsections. Subsection 2.3.1 provides an overview of our proposed architecture and the data flow in the architecture. Subsections 2.3.2-2.3.4 discuss the details of techniques used in the architecture according to the data flow.

2.3.1 Overview

The inputs of the system are the intra prediction mode, size of TB, residuals of TB and its coordinates, xTb and yTb. The outputs are the reconstructed pixels.

The data flows within the system and the architecture are shown in 0. The entire process is executed in pipeline and includes four steps. These steps correspond to the stages R, D, P and W. First, in the R stage, the ad- dress and reference pixels' position based on the prediction mode and coordinates of Tb are generated. Second, in the D stage, reference pixels are fetched from the memory and reference pixel substitution is proceeded before storage in registers. Third, in the P stage, reference pixels are filtered then used for prediction and reconstruction of the pixels based on 4×4 block. Finally, in the W stage, reconstructed pixels are written back to the memory. The following are highlighted: LUT-RSFS (R stage), 4×4 Prediction Block (P stage), and HBRDF (W stage).

Table 2 The selection of reference samples and SRAM banks, given (X,Y)=(0,3), and 16×16 TU (Bold text represents the samples for filtering)

Prediction mode	**Reference samples selected by LUT-RSPG**	**SRAM banks**

23	left:14,13,12,11,10,9,7,6,5,0; top-left(TL);
	top:0,1;
	top-left(TL); top:0,1;
		0,4,5,6,7,TL

28	top:1,2,3,4,5,6,7,8	0,1,2
Planar	left:15,14,13,12,11,10;TL; top:0,1,2,3,4 top:0,1,2,3,4	0,1,3,4,5,TL
DC	Left: 11,12,13,14,15,16	3,4,5

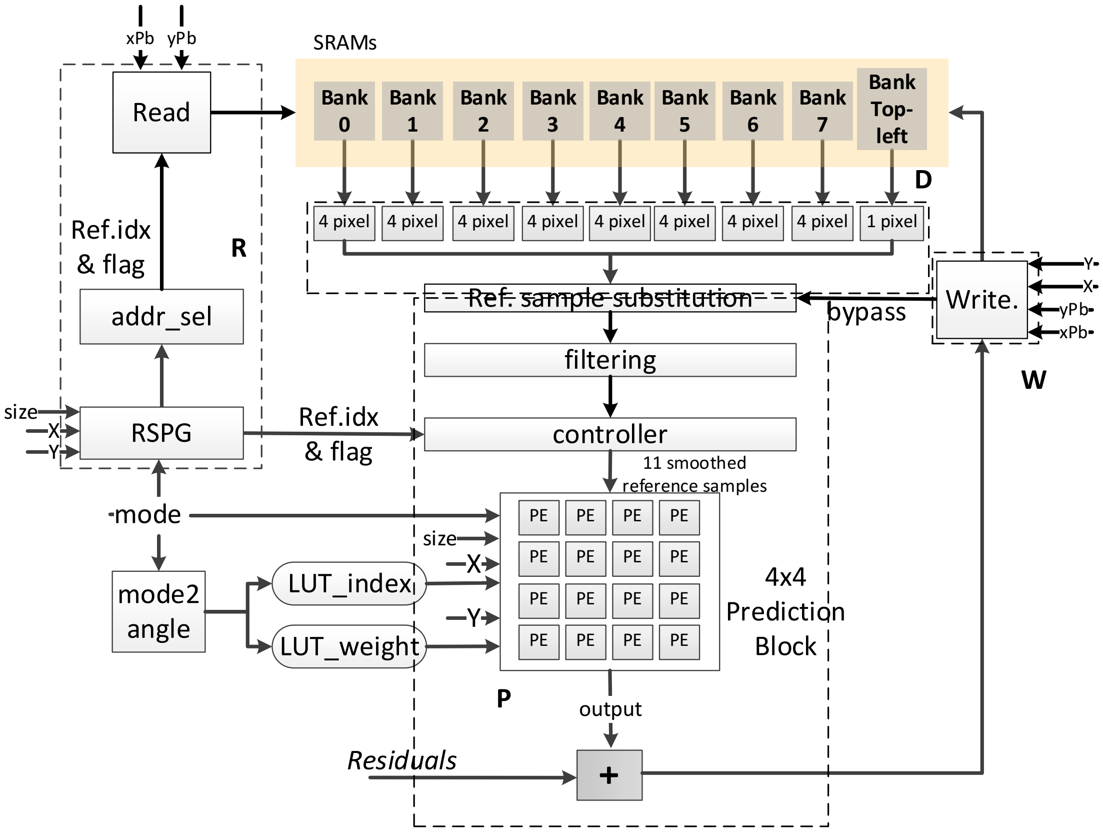


Fig. 10 The architecture of the intra prediction

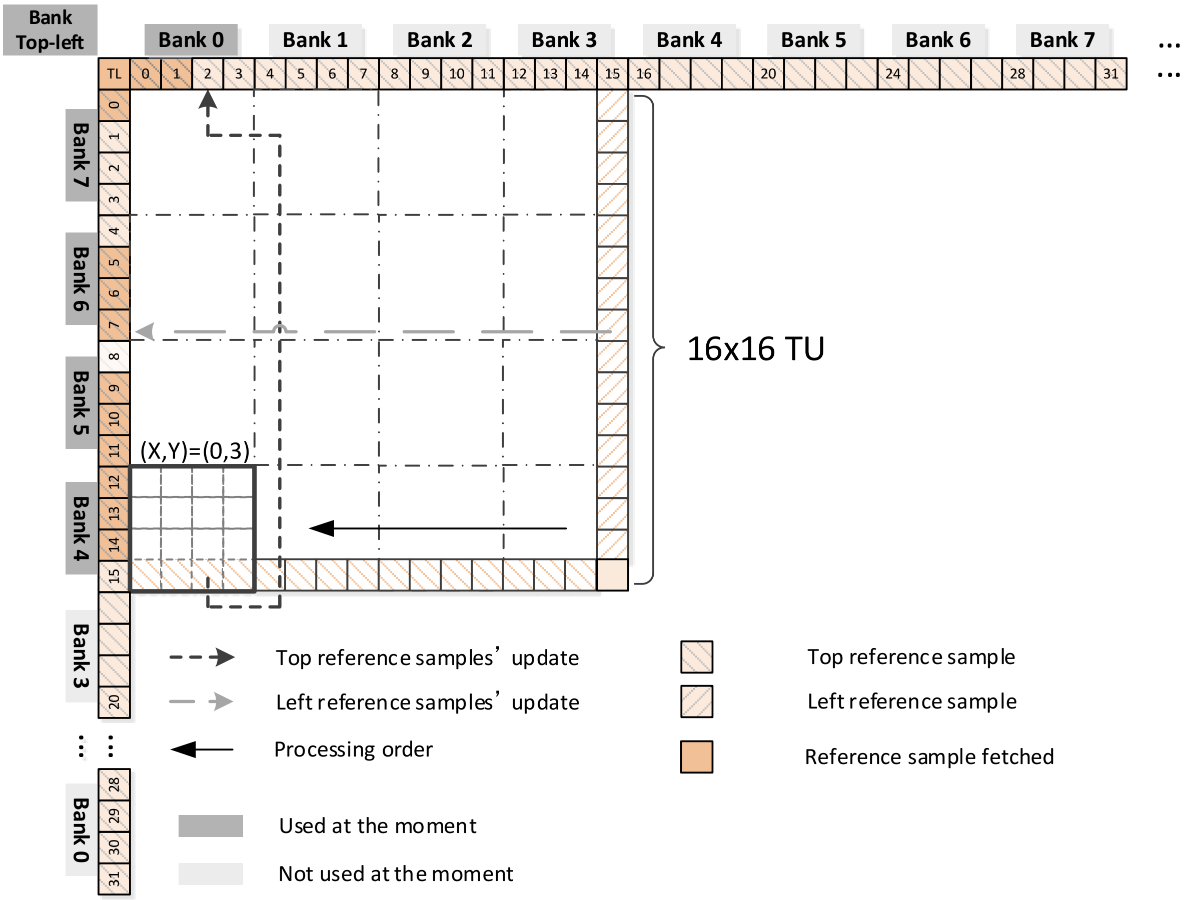


Fig. 11 An example to show how RSFS work.
Given prediction mode is 23, (X,Y)=(0,3). The reference samples update, 4×4 block processing order and arrangement of 8 cyclic banks are also shown.

2.3.2 LUT-RSFG

Reference sample fetching is an important issue in intra pre- diction architecture. It has a great impact on the system performance. As previously mentioned, we propose a new reference samples fetching scheme. The main idea is to break a TU into smaller parts before processing, so that less reference samples are fetched each time. The new scheme reduces the maximum number of fetched reference pixels from 99 (as used in [11]) to 13, improving performances and utilizing hardware resources. The scheme includes two parts: The LUT based Reference Sample Position Generator (LUT-RSPG) and the 8 Cyclic Static Random Access Memory (SRAM) banks. In the follows, we present our scheme first and use an example, combining the Table 2 and Fig. 11.

2.3.2.1 LUT-RSPG

Reference samples are fetched, smoothed, and then used for prediction. Reference sample fetching gets complicated in H.265 with the increased TU's size, prediction modes, and smoothing methods. Angular prediction modes 11 to 25 need some discontinuous reference samples and planar mode needs additional top-right and bottom-left samples for prediction. Moreover, for smoothing, some additional samples are also required.

I develop the RSPG based on LUT for reference samples fetching. As shown in 0, given the size of TU, prediction mode, and coordinates X,Y of 4×4 blocks in TU, RSPG indicates which reference samples are needed for prediction. The position of reference samples can be found in the LUT by getting the indices and the flag that indicates the type of reference samples–top, left, or top-left. The second column in Table 2 The selection of reference samples and SRAM banks, given (X,Y)=(0,3), and 16×16 TU (**Bold** text represents the samples for filtering) illustrates the reference samples to fetch in some given prediction modes, and the bold number represents the samples used for filtering. I use LUT to substitute complex computation in hardware, such as the combination of multiplications and additions, so that a lower cost in hardware and higher performance can be achieved. Since the LUT used for the fetching scheme has numerous items, we used software to automatically generate the LUT. By modifying the HEVC Test Model (HM), we record the relative locations of reference samples and the corresponding memory banks, for all 4x4 positions under each of all prediction modes.

2.3.2.2 8 Cyclic SRAM banks

After LUT-RSPG decides which reference samples to be fetched, we must ensure that at any time the required reference samples can be fetched from the memory in one clock cycle. Since an SRAM bank allows the reading of only one of its cell's data by a specific address per cycle, collisions may occur if some reference samples to be used are stored in the same SRAM bank but in a different cell. For instance, in 0, the top samples 0–3 in bank 0 and left samples 28–31 in bank 0 cannot be fetched in one clock cycle.

To avoid collisions, we developed 8 Cyclic SRAM Banks in order to store the reference samples. “Cyclic” and “8 banks” are two key points. The arrangement of reference samples stored in SRAM banks is shown in 0. The cyclic order guarantees that any neighboring 32 samples can be fetched at the same cycle. Especially, for prediction modes 11 to 25, the top reference samples 0–4 and the left reference samples 0–12 have to be fetched simultaneously. In this case, data in bank 4 5 6 7 and in bank 0 1 is fetched in one cycle. “Cyclic” is designed to solve this problem.

The number of SRAM banks is another issue to discuss. I want the smallest number of banks, provided that no collisions occur, because a larger number of SRAM banks makes the chip’s area and power consumption larger. Further, the number of banks would be more adequate as a power of two for high performance; this is because the ad- dress can then be calculated by shift and the add operation, instead of division. For our work, we found that 8 is the most suitable number of SRAM banks.

After detailing the LUT-RSPG and 8 Cyclic SRAM banks, we use an example to show how the Parallel Reference Sample Fetching Scheme works. The graph and data can be referred in 0 and Table 2 The selection of reference samples and SRAM banks, given (X,Y)=(0,3), and 16×16 TU (**Bold** text represents the samples for filtering), respectively. For instance, when the TU’s size is 16×16, prediction mode is 23, and current processing 4×4 block (X==0, Y==3) is located in bottom-left corner inside the 16×16 TU, then the smoothed reference samples required are left 13,10,6, 3, top-left samples, top 0. As two-tap [1 2 1] FIR filtering is needed in 16×16 TUs, the neighboring left and right samples of 12, 13 and 14 need to be fetched from the SRAMs. Thus, at this cycle, reference samples in SRAM banks 4, 5, 6, 7, Top-left, and 0 are used.

Top and left reference samples’ memory deployment is shown in Table 3. Top and Left reference samples share the same 8 banks (For each bank, 252 words × 32 bits); while the top-left samples uses one bank (2048 words × 8 bits). Each of the eight banks is divided two parts, Top (line buffer, C, E) and Left (B, D, B’ and D’). Each parts is divided into several regions, allocated to different addresses.

There are two types of operation–read and write. Reading occurs at each clock cycle; while writing occurs when current processing 4×4 block locates on the right or bottom boundary of a TU. At this time, the rightmost and bottommost reconstructed samples are written into memory for the prediction of neighboring TU. The address for reading and writing is decided by the coordinates of the TU. For in- stance, it reads from line buffer if the TU is located on top- most of a Coding Tree Unit (CTU). Besides, we use Ping- Pong buffering to prevent the data to be read, from being covered by the newly written data. That’s why we divide the memory into several regions.

Table 3 SRAM Deployment for top, left and top-left luma samples

Storage type	**Name**	**Region**	**Addr.**	**Bit depth**	**Pcs.**	**Bytes**
1R1W SRAM	Top & Left Top & Left	Line buffer	0-239	32	8	8064
		C & E	240-243			
		B & D	244-247			
		B′ & D′	248-251			
	Top-left		0-2047	8	1	2048
	Total	10112				

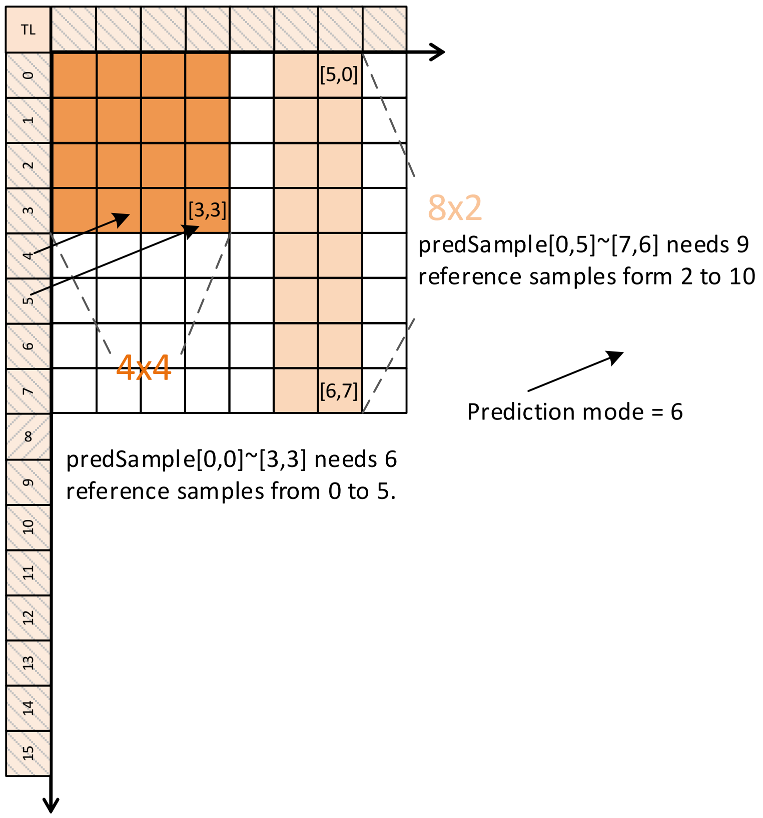

TL
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
[5,0]
[3,3]
[6,7]
8x2
4x4
predSample[0,5]~[7,6] needs 9 reference samples form 2 to 10
Prediction mode = 6
predSample[0,0]~[3,3] needs 6 reference samples from 0 to 5.

Fig. 12 Analysis of the required number of reference samples for prediction of a 4×4 block
(intra prediction mode is 6).

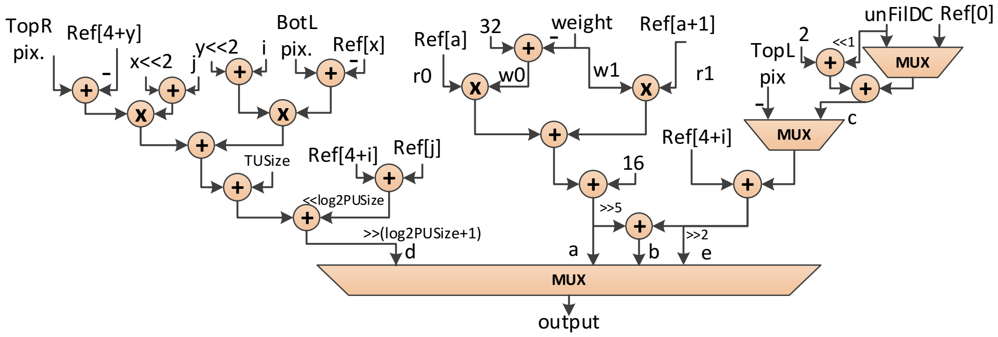


Fig. 13 The structure of one PE, designed for Planar, DC and angular predictions.

2.3.3 4×4 Block Based Prediction

First, this section presents the reason of using 4×4 blocks. Then, the details and the originality of the 4×4 blocks usage are exposed.

To achieve 8K, 4320p@120fps, we have to satisfy (5), where f represents the clock frequency of the system (cycles/second) and n is the number of pixels processed by the system per cycle.

$$7680 * 4320 * 120 = f * n \tag{5}$$

For a practical system, we aim at decreasing f in order to reduce relatively the power consumption. For a system processing at 16 pixels per cycle, when luma/chroma samples are processed in parallel, the required clock frequency is 238 MHz. If the system processes 32 pixels per cycle, the required clock frequency is 119 MHz

If our system process 32 pixels per cycle, half of the processing units are wasted, since the minimum TU size is 4×4. For the 16-pixels-per-cycle's processing block, it could be 4×4 square block, the 8×2, or 16×1 rectangle blocks. I found 4×4 block is the best. Because it requires the least numbers of reference samples in worst cases. I use prediction mode 30 as an example, as shown in 0. Further- more, 4×4 blocks can enhance the utilization of the hardware resources, since all sizes of TU can be divided into one or more 4×4 blocks.

The 4×4 prediction block we proposed consists of 16 Prediction Elements (PE). Each of it processes one sample per cycle. The inputs of each PE inside the 4×4 block are prediction mode, weight, and reference index. The detailed design of each PE is shown in 0. I generate 2 LUTs to get the weight and reference index instead of using formula to calculate in hardware, in order to reduce the path delay. In each predictor, there are 4 multipliers, 10 adders, and 5 multiplexers. For the predictor in first row and column, 4 additional adders and 2 multiplexers are required for filtering by each predictor. "a" is the output of angular prediction mode; "d" is the output of Planar; "e" and "b" are the output for boundary samples smoothing in DC mode and horizontal, vertical modes correspondingly, when TU is 4×4 to 16×16. The hardware used for boundary samples smoothing are only in the first row and first column in the prediction block. It should be noted that, from our synthesis result, we find that the data path is 0.3 ns shorter if Planar mode and angular mode do not share one multiplier. Thus we have the planar and angular modes not share one multiplier to achieve a higher performance.

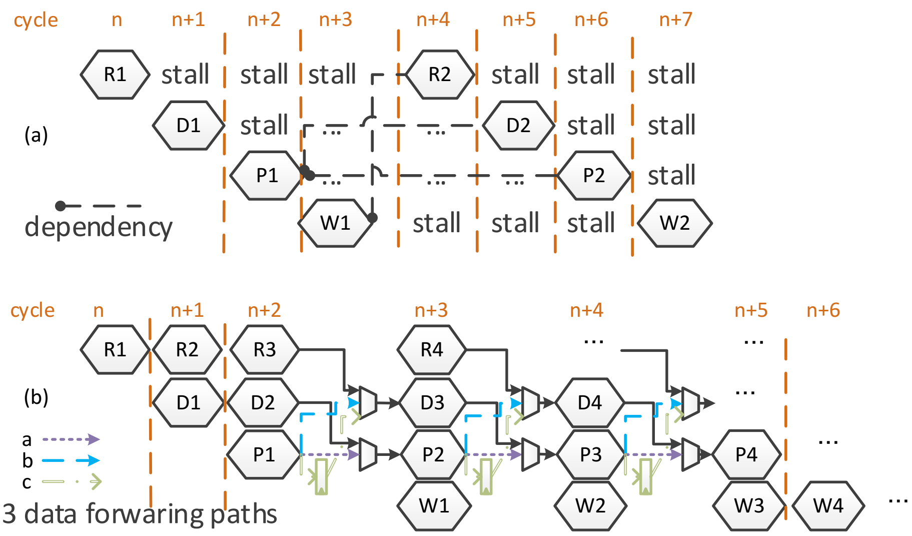


Fig. 14 Dependency problem in blocks

(a) Stall of 3 cycles to solve the dependency problem in a 4×4 TU

(b) Use of Data Forwarding to solve the dependency problem in 4, 8 TU

R1: Read reference samples for PB1.

D1: Get data from SRAM and reference sample substitution for PB1.

P1: Calculate predicted samples and add residuals.

W1: Write back the result.

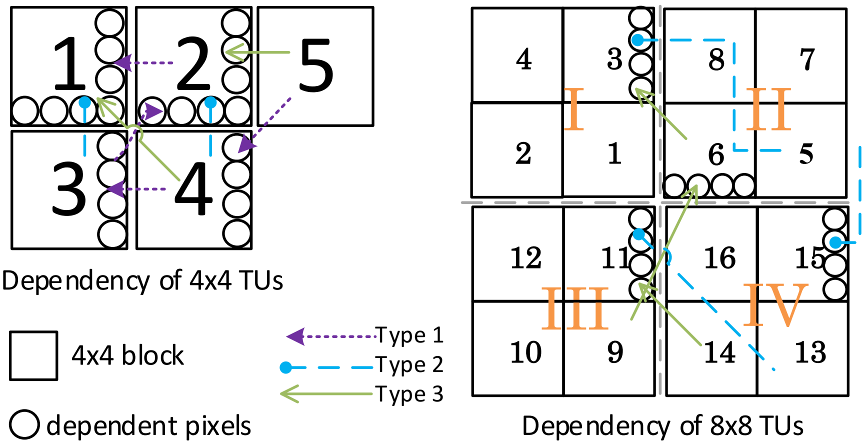


Fig. 15 Three dependency cases in 4×4 TUs and 8×8 TUs

2.3.4 Hybrid Block Reordering and Data Forwarding

This section exposes the data dependency issue in intra prediction of 8K application and shows how our proposed solution, called Hybrid Block Reordering and Data Forwarding, solves this problem.

As we know, prediction needs the neighboring TU's reconstructed pixels as reference pixels, which are stored in the memory. This need illustrates the data dependency between TUs. The latter issue occurs when the TU needs the reference pixels, not yet stored in the memory. However, at low system throughput, such as 4K@30fps in [11], this problem does not occur. Because more cycles are required to process a TU, when neighboring pixels are needed, they are already stored in the memory.

I will show how Block Reordering and Data Forwarding jointly solve the data dependency problem. Block Reordering is applied to 8–32 TUs. Because an 8, 16, and a 32 TU can be divided into 4, 16 and 64 4×4 blocks, respectively, we can arrange the process order such that the pixels that will be used as reference pixels are processed and stored into the memory earlier. I begin with the right-bottom 4×4 block, process from right to left, from down to up, and end on the top-left block. The processing order is illustrated by curve in 0. This method can solve the dependency problem if the previous TU is 16×16 or 32×32. If the previous TU is 8×8 or 4×4, we need to use the following method below, called Data Forwarding.

Data Forwarding is widely used in hardware design. The idea in our design considers using some registers to temporally store the reconstructed pixels with dependency, and when these pixels are needed, we control the reference samples to be fetched from these registers, instead of from the memory. The novelty is how to apply this technique to solve the dependency problem, especially to determine whether the reference pixels should be fetched from the registers or from the memory.

There are three types of dependency in this work. The first one is that P2 depends on P1, in 0 (a). Since the 4×4 prediction block processes a 4×4 TU in a cycle. As shown in 0 (b), PB1 is processed at cycle n+2 at the P1 stage, and PB2 in Z-order is processed at cycle n+3 at the P2 stage. At cycle n+3, PB2 may need the reference samples located in PB1, while the result of PB1 cannot yet be used by PB2 since it is being written back to the SRAM. If such reference samples are read from the SRAM and used by PB2, then the obtained results are erroneous because the reference samples fetched from the SRAMs are not the expected reconstructed samples of PB2. To solve this problem, we build the Data Forwarding path ‘a’, shown in 0 (b), to send forward the result of P stage to the registers at cycle n+2, and at cycle n+3, the following TU reads the reference samples from the corresponding registers, instead of waiting for the predicted samples to be written into the SRAMs.

The second type is that D2 depends on P1, in 0 (a). When $\mathrm{PB_k}$, writes back the reconstructed samples in $\mathrm{W_k}$ stage while $\mathrm{PB_{k+2}}$ use the data to do substitution in the $\mathrm{D_{k+2}}$ stage. For this type, we build the Data Forwarding path ‘b’. The last type is that R2 depends on W1, in 5(a). For 1R1W SRAMs, the reading address and writing address cannot be the same when writing and reading operations happen at the same cycle. Reading has to wait until writing finishes. When $\mathrm{PB_k}$ try to write back the reconstructed samples in $\mathrm{W_k}$ stage while $\mathrm{PB_{k+3}}$ reads the same data from the same address, the error occurs. To solve it, we build the Data Forwarding path ‘Type 3’, shown in 0 (b). It keep writing and does not read. When it needs data, it reads from path ‘c’ directly.

Overall, Three Data Forwarding paths, one from stage $\mathrm{P_k}$ to $\mathrm{P_{k+1}}$ (path a), one from stage $\mathrm{P_k}$ to $\mathrm{D_{k+2}}$ (path b) and one from stage $\mathrm{W_k}$ to $\mathrm{D_{k+2}}$ (path c), are built to eliminate three types of dependency respectively. I use 0 to shows how the dependency occur in the 4×4 TUs and 8×8 TUs. For instance, the $\mathrm{PB_5}$ in $\mathrm{TU_{II}}$may use the results of $\mathrm{PB_3}$ in $\mathrm{TU_I}$, and the $\mathrm{PB_{13}}$ in $\mathrm{TU_{IV}}$ may use the results of $\mathrm{PB_{11}}$ in $\mathrm{TU_{III}}$ as reference samples. This is the second types illustrated above, which is marked by blue dash line.

As shown in 5(a), we know the stalling of pipeline for three cycles could eliminate the dependency, at the expense of throughput. In our technique, no stall occurs in our method, such that the throughput is significantly increased. Even though Data Forwarding can solve the dependency problem of all other sizes besides 4×4 and 8×8, we do not apply it elsewhere. Because data forwarding makes the pipeline design and implementation more complicated and the critical delay becomes longer. Furthermore, since the processing order inside a TU is not restricted, then alternatively we can use processing reordering to solve this problem. This method is easier compared to Data

Forwarding.

2.4 Implementation Results

The proposed architecture is designed in Verilog, and synthesized in SMIC 40 nm standard-cell libraries. The layout is shown in 0. Before the layout, the delay of the critical path is 3.65ns, and after the layout, it is 3.8ns. If the luma and chroma modules work in parallel, the throughput's requirement, 4320p@120fps, can be achieved when system's frequency reaches 260MHz. For 16×16 and 32×32, one and four additional cycles are required for data preparation respectively. Since an overhead of 1/17 of the total processing time used for data preparation, it's required to achieve the target by reaching 238*17/16=253MHz in the worst case. Hence, including the time for data preparation, the through- put is 16*(16/17) = 15.1 pixels/cycle or 22.5 samples/cycle in 4:2:0 format. A 4/8/16/32/64 PU can be processed in 1/4/17/68/272 cycles, as given in Table 4.

Table 4 Worst throughput cases and processing speeds

N	Substitution & filtering (Cycles)	Perdition (Cycles)	Throughput (Samples/ Cycle)	Time to complete a TU (Cycles)
4	0	1	16	1
8	0	4	16	4
16	1	16	15.1	17
32	4	64	15.1	68

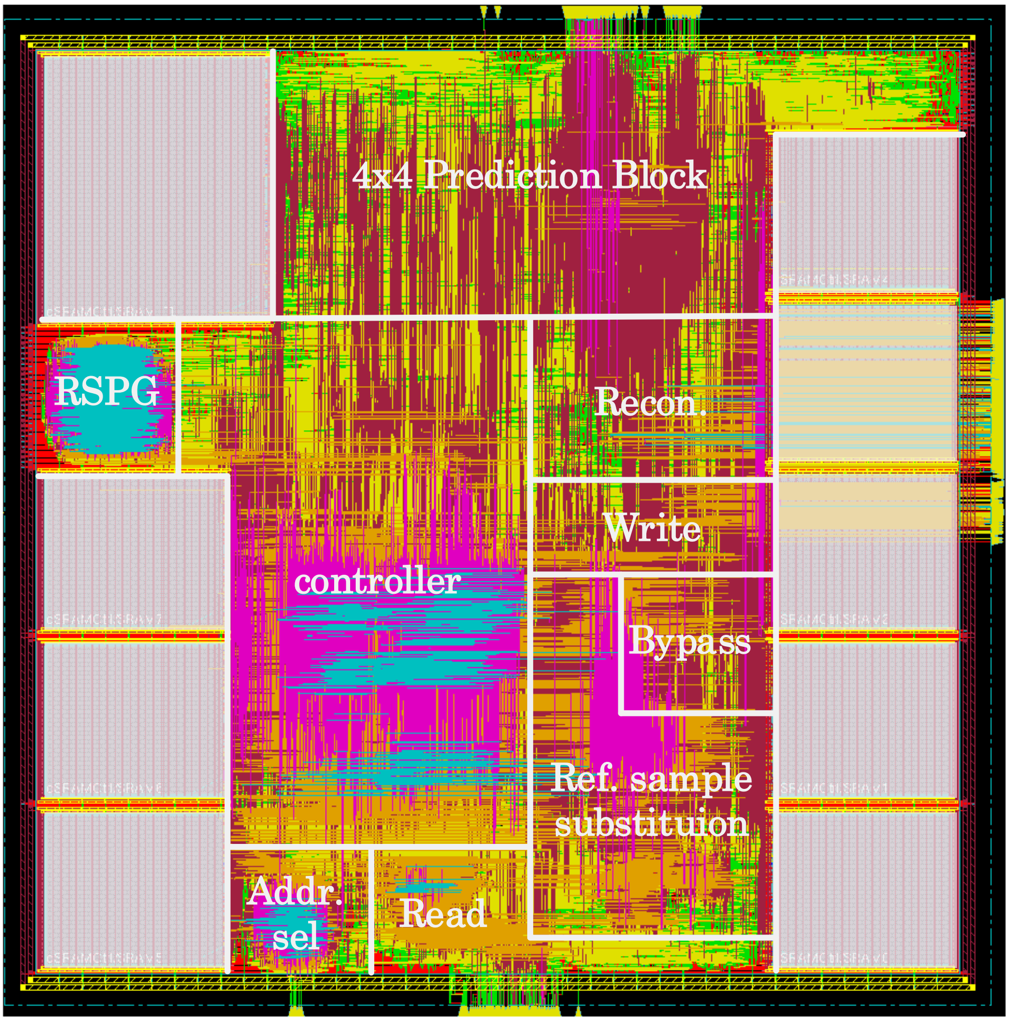


Fig. 16 Layout of the proposed intra architecture.
The luma module (10-bit) with a core size of 0.36*mm*^{2} and chip-area utilization of 80%. The location of each module are marked, while the SRAMs for neighboring pixels and line-buffer are shown in white regions.

Table 5 Comparison of H.265/HEVC intra prediction architectures for video application

	Proposed architecture		**Huang'[6] 2013**	**Jung'[7] 2013**
Platform	40 nm		40 nm	130 nm
Bit depth	8-bit	10-bit	8-bit	8-bit
Area (gates)	212K	252K	27K	41K*
SRAM	430B	538B	612B	N/A
Line buffer	20KB	25KB	16KB**	N/A
Pred. Mode	All			
PU sizes	All			
Specification	4320p, 120fps @260MHz		2160p, 30fps @200MHz	N/A N/A
Min. Tp (samples/cycle)	22.5		2	8
Norm.TP (samples/cycle/k- gate)	0.103	0.090	0.074	0.198
Norm. rea Complexity (gates)	212K	252K	304K	N/A***

By taking the line buffer into account, the size of the on-chip memory used for either 8-bit luma samples or chroma samples is 10K, as shown in Table 2. Thus, the overall size is 20 KB.

The comparison with other works is shown in Table 5. As many key functions are not included in [12], the comparison with it is not actually fair. Compared with [11], the throughput of our design is 16x higher, with logic area only 7.85x more. In our design, the normalized throughput is 0.103, and the normalized area complexity is 212K, compared with 0.074 and 304K in [11], respectively. Thus, a higher normalized throughput or a lower normalized area complexity is achieved by the proposed design. The improvement comes mainly from two aspects.

System's high throughput is achieved by following aspects: First, by breaking a TU into smaller blocks, we reduce the number of reference samples fetched and multiplexed for prediction. Second, we separate the multi- pliers for the prediction of the Planar mode and angular modes. These first two aspects make data path shorter in order to achieve a higher performance. Third, we develop LUT-RSFS to arrange a tight schedule for reference samples preparation and prediction, 16/17 (approximately 94%) of cycles are utilized for intra prediction. Finally, we use Block Reordering and Data Forwarding to get rid of the data dependency, so that the system can operate under high throughput.

Besides aiming at a high throughput, we also aim at reducing the circuit area. First, as mentioned above, we break TU into small blocks and devised LUT-RSFS to fetch reference samples precisely. They reduced number of the reference samples fetched for prediction in the worst case, from 99 to 13 reference samples, so that the complexity of the control circuit for selecting the reference samples is decreased, and the area of relevant circuit can also be reduced. Second, we select 4×4 block as a prediction unit. Since it is adaptive to all TU sizes, hardware resource is saved. Finally, more reference samples can be reused by the 4×4 block and more neighboring predicted samples share probably the same reference samples, so that the area of circuit gets further reduced.

2.5 Summary

In this paper, we have presented an 8-bit/10-bit adaptive intra prediction hardware architecture for H.265 4320p@120fps application. Based on the divide-and-conquer strategy, we proposed two techniques. Using the LUT-RSFS, required reference samples are fetched from the SRAMs at each cycle with low complexity and small circuit area. By exploiting Block Reordering and Hybrid Data Forwarding, we have minimized the idle time and eliminated the dependency between TUs in order to increase the throughput. Hardware utilization of 94% is achieved and only 272 cycles are used to process a 64×64 block in worst case. The demerit of the proposal is that the bandwidth of SRAM in increased, as multiple loadings of reference samples are required for a block larger than 4x4. This would be a problem to solve in the future work.

3. Dual-clock VLSI architecture of HEVC Sample Adaptive Offset Estimation

3.1 Introduction

Compared to the previous H.264/AVC standard [21], H.265/HEVC doubles coding efficiency by employing a number of new coding tools. Especially, an Sample Adaptive Offset (SAO) component is newly introduced as one of the in-loop filters (ILF) ,which contributes to up to 18% BD-rate reduction [22]. In H.264/AVC, Deblocking Filter (DBF) [23] is the only ILF. Its VLSI implementation has been discussed in many previous works [24] [25] [26] [27]. In H.265/HEVC, DBF [28] has been simplified [29] [30] [31] [32] [33] [34] and SAO dominates the complexity of ILF especially in a video encoder. Several previous works discussed SAO's implementation. Joo, et al. [35] [36] proposed to utilize the intra prediction mode to predict the Edge Offset (EO) type, so that the number of EO types could be reduced to save the encoding time. Choi et al., [37] evaluated several algorithm-level improvements for SAO. Gao et al., [38] developed a low complexity SAO algorithm based on class combination, band offset (BO) pre-decision and merge separation category. Rediess et al., discussed the architectures of statistics collection and parameter decision, two main components of SAO, in [39] and [40], respectively. Mody et al., [41] designed an SAO estimation architecture supporting 4K@60fps encoding. Zhu et al. [42] [43] developed a fast SAO estimation algorithm its VLSI architecture supporting 8K encoding. The complete implementations of SAO estimation [41] and [42] [43] both require relatively large circuit area, which still has plenty room for improvement.

This work aims at designing an efficient VLSI architecture of SAO estimation in H.265/HEVC. To achieve high area efficiency, we propose three techniques:

- Dual-clock SAO architecture: The highly heterogeneous data flow of statistics collection (SC) and parameter decision (PD) in SAO causes each part to require a completely different preference in working frequency. Such a different preference is the main obstacle for an efficient implementation. This technique addresses heterogeneous data flow by separately driving SC and PD at a high- and a low-speed clocks, respectively, so that each part could be integrated together efficiently. It reduce the overall area by 56%, from 156K to 68K gates.

- Coarse range selection for BO (CRS): Based on the analysis of band distributions in each Coding Tree Block (CTB), and on hardware resources for finding best bands, this technique estimates the range of bands before SC with an accuracy of 60-80% and shrinks the range of fine processed bands 32 to 8 to reduce the circuit area.
- Accumulator bit width reduction (ABR): By exploiting the mutual exclusion relationship among categories/bands existed in the accumulation process in SC, this technique carries out an early termination to accumulators reaching a threshold, to further reduce their circuit area.

The proposed VLSI implementation employing the above techniques occupies 51K logic gates, which is only one-third of circuit size of [42], at the same throughput and comparable coding efficiency. With a high-speed clock of 1.3 GHz and a low- speed clock of 217 MHz, 8K@120fps SAO real-time encoding can be achieved.

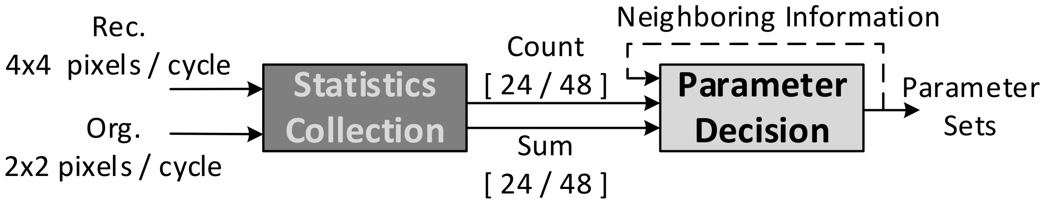


Fig. 17 The overview of SAO.
Details of the proposed statistics collection engine and parameter decision engine are in Fig. 22 and Fig. 23, respectively.

The rest of paper is organized as follows. Section 3.2 gives an introduction to SAO in H.265, its data flow and several hardware friendly approaches for the design. Section 3.3 analyzes the data flow of SAO, the key challenge and introduces the first proposed technique. Section 3.4 A analyzes the characteristics of BO and introduces our second proposed technique utilizing those characteristics. Section 3.4.1 explains how mutual exclusion among categories/bands generate inefficiency in hardware utilization, and give our third proposed technique. Section 0 shows implementing results and gives some analysis on the performance of the proposed design, followed by the conclusion in Section 3.6.

3.2 SAO algorithm

SAO aims at reducing the distortion of the reconstructed pictures, by adaptively adding offsets to the reconstructed samples at both encoder and decoder. The SAO parameters, i.e. how the offsets should be generated and applied, are signaled at the Coding Tree Unit (CTU) level. The offset to be applied depends on the classification of the target sample. There are two kinds of classifiers: Edge Offset (EO) and Band Offset (BO). The sample classification of EO depends on the comparison between the current sample and its neighboring ones, while the sample classification of BO depends on the value of current sample itself. The optimal classifier and offsets for each CTU is found during the encoding process, called SAO estimation, which comprises the SC and PD phases, as shown in Fig. 17. In SC, the BO and EO classifiers classify each reconstructed sample in a CTU into different bands and categories, respectively. The classification statistics of the current CTU are collected. In PD, based on the statistics and the neighboring (left and upper) SAO parameters, the optimal parameter sets achieving the lowest rate-distortion cost are found. The parameter sets include the SAO mode, types and offsets, as shown in Table 6.

Table 6 Output (parameter sets) of SAO.
Luma and chroma share the same set of parameters in gray parts. Cb and Cr always share the same mode and type.

SAO Mode	SAO Type	SAO Offset	Type AuxInfo
OFF: 0	N/A	N/A	N/A
New Mode: 1	EO 0: 0	Offset [0:3]	
	EO 90: 1		
	EO 135: 2		
	EO 45: 3		
	BO: 4		0-28
Merge: 2	Merge Upper: 0	Follow merged CTU	
	Merge Left: 1		

3.2.1 Statistics Collection

For EO, the category of each sample is decided according to its relationship with neighboring samples, following 4 patterns, the horizontal (EO 0), the vertical (EO 90) and two diagonal (EO 45 and EO 135) directions, as shown in Fig. 18. A sample that falls into none of these categories is classified into category 0.

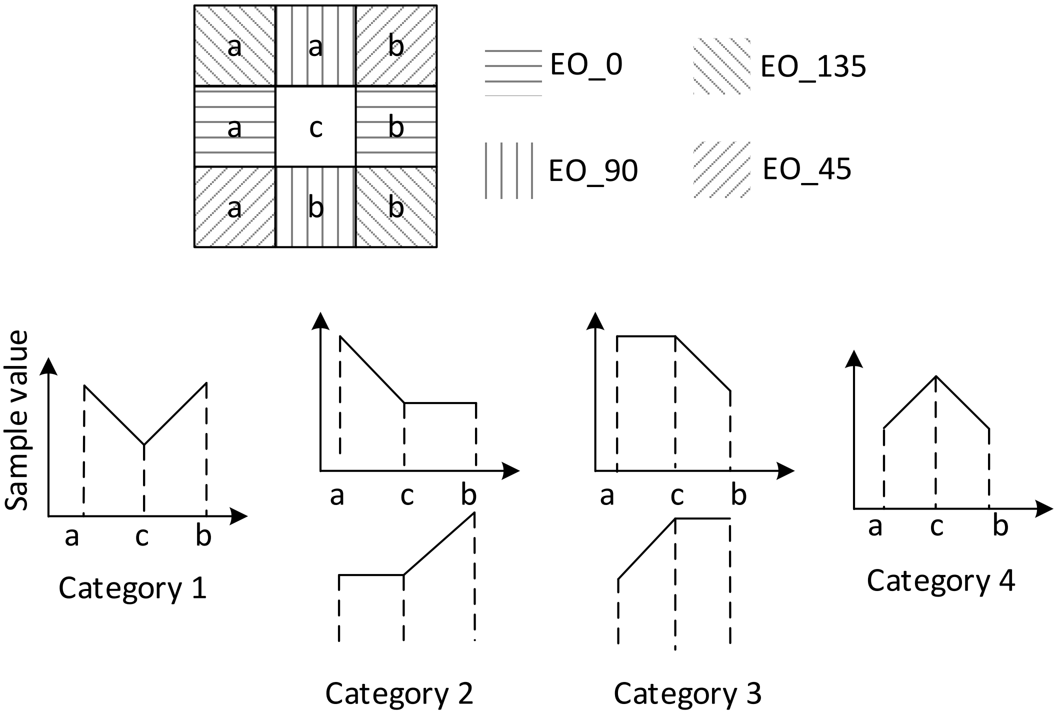


Fig. 18 EO patterns and categories.

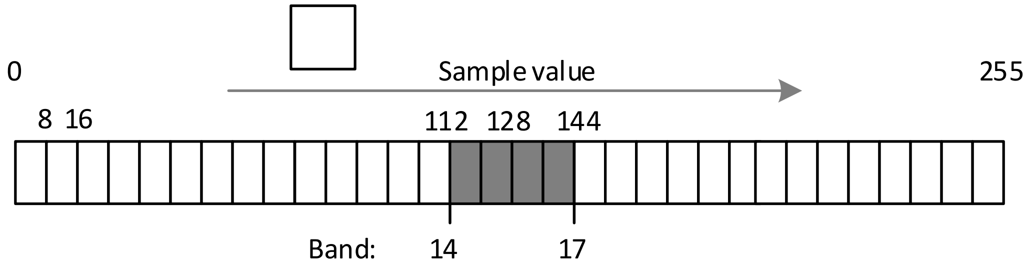


Fig. 19 Thirty-two Bands division.
The value of an 8-bit sample (dynamic range (0 to 2BitDepth−1)) is evenly divided into 32 bands. The best consecutive four bands are chosen as candidates, e.g, Band 14 to 17.

For BO, the band of a sample is decided according to the value range it falls in. The entire dynamic range (0 to 2BitDepth−1) is evenly divided into 32 bands, as shown in Fig. 19. An 8-bit sample is classified into band K if it ranges from 8K to 8K+7. Based on the statistics collected within a CTB, the best consecutive four bands and their corresponding offsets are chosen as candidates to compare with the EO patterns.

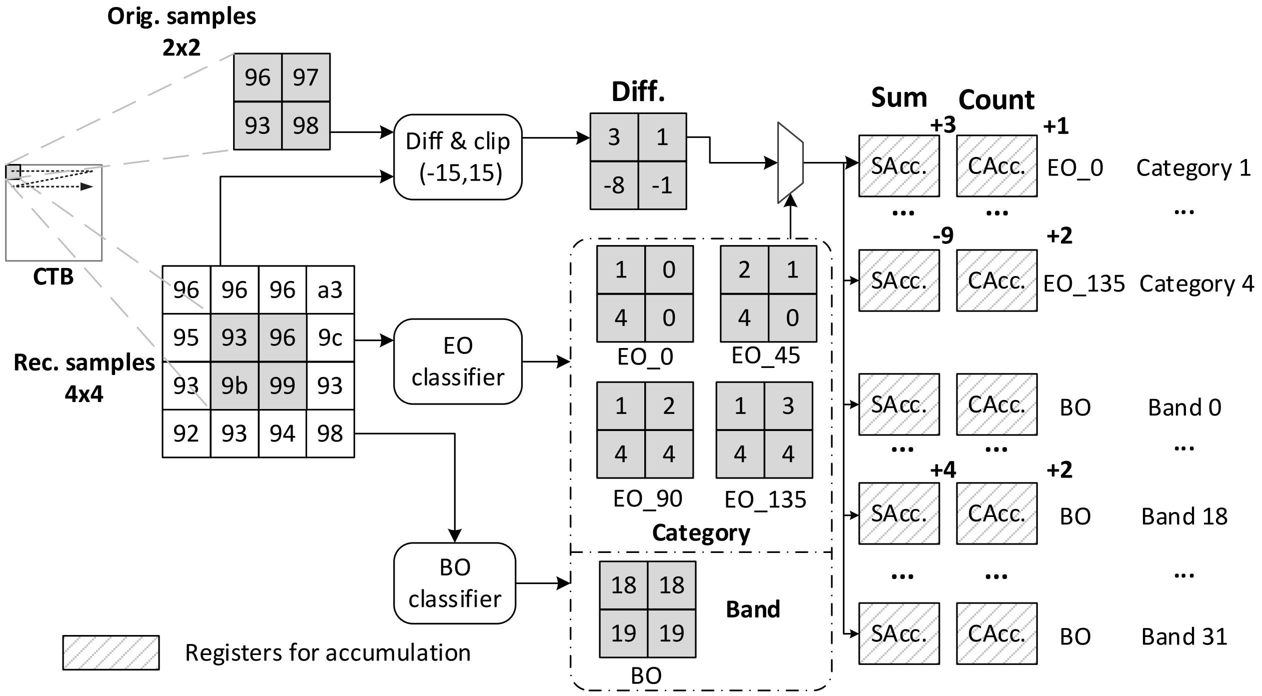


Fig. 20 An example to illustrate the statistics collection process.
SAcc and CAcc are abbreviations for accumulators of Sum and Count. E.g. BO classification is performed to the 2x2 reconstruction samples (0X93,0x96,0x9b,0x99). Since the first two samples belong to band 12, the differences (Org.-Rec.) belong to band 12 are summed up. The SAcc and CAcc of band 12 add to 4 and 2, respectively.

3.2.2 Parameter Decision

The purpose of PD is to decide the parameters for the current CTU, based on the statistics collected in SC. The parameters include the SAO mode, SAO type, auxiliary in- formation and four offsets. The possible outputs are listed in Table 6. The set of parameters with lowest rate-distortion (RD) cost is chosen as the one to be coded. RD cost is defined as:

$$\mathrm{Cost} = \mathrm{D} + \lambda * \mathrm{rate} \qquad (6)$$

,where rate is the number of bits to code the parameters and λ is the Lagrange multiplier. Distortion between the original and reconstructed samples modified by SAO can be described by the following equation:

$$\mathrm{D_{post}} = \sum_{c \in CTB} \left(\mathrm{org(c)} - \mathrm{rec_{post}(c)}\right)^2$$

$$= \sum_{c \in CTB} \left(\mathrm{org(c)} - (\mathrm{rec_{pre}(c)} + offset)\right)^2 \qquad (7)$$

where the offset is calculated by the Sum and Count from SC

$$\mathrm{offset} = \frac{\mathrm{Sum}}{Count.} \qquad (8)$$

By evaluating all the bands, EO patterns, and the merge candidates, the parameter sets with minimum cost are chosen as the final decision. The comparison of distortions can be simplified by eliminating the org and rec in D and D_{post} as the following equations. The details can be referred in [22]

$$\delta D = D_{post} - D_{pre} = Count * offset^2 - 2 * offset * Sum \qquad (9)$$

3.2.3 Hardware friendly SAO

To improve the algorithm’s friendliness to hardware implementation, a previous work [42] made modifications to the original SAO algorithm in the HM reference software, as listed in the Table 7 from Mods 1 to 4.

Table 7 Comparison of modifications made on the HM-16.0

<table>
<tr><th>No</th><th>Name</th><th>HM-16.0</th><th>ICIP [25]</th><th>Props.</th></tr>
<tr><td>1</td><td>Number of iteration for offsets</td><td>At most 7 iteration</td><td colspan="2">No iteration</td></tr>
<tr><td>2</td><td>Evaluation method for best band</td><td>RD-Cost</td><td colspan="2">Distortion</td></tr>
<tr><td>3</td><td>Rate</td><td>CABAC</td><td colspan="2">constant probability model[25]</td></tr>
<tr><td>4</td><td>Normalization of RD-cost</td><td>The RD-cost of New RDO mode</td><td colspan="2">The RD-cost of Merge mode</td></tr>
<tr><td>5</td><td>The range of difference</td><td>[-1023,1023]</td><td>[-7,7]</td><td>[-15,15]</td></tr>
<tr><td>6</td><td>Samples unused in SC (Fig. 21)</td><td>Region A</td><td>Region A∪B</td><td>Region A∪C</td></tr>
<tr><td>7</td><td>Number of bands</td><td>32</td><td>32</td><td>8</td></tr>
<tr><td>8</td><td>Accumulator bit width reduction</td><td>---</td><td>---</td><td>Accumulator terminates when reaching a threshold</td></tr>
</table>

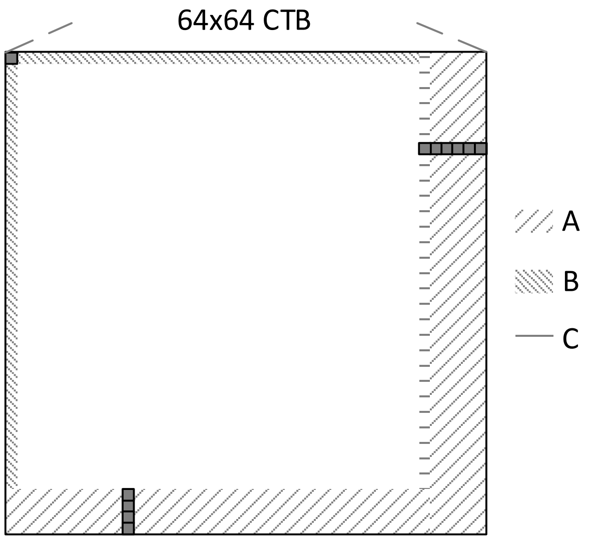


Fig. 21 Samples not used in statistics collection, referring to No.6 in Table 7

In this work, we further apply Mods 5 to 8. Mods 5 enlarges the range of difference between original samples and reconstructed samples for a higher precision. Mods 6 utilizes the top and left boundary samples for statistics collection to increase calculation accuracy. As Mod 7 we proposed a technique to estimate the most probably selected bands in a CTB and to reduce the searching space for the best bands from 32 to 8 bands, the detail of which is explained in Section 3.4.1. As Mod 8 we propose a bit width reduction technique for the accumulators, with details given in Section 3.4.2.

3.3 Dual-clock Architecture

3.3.1 Heterogeneous data flows of SC and PD

The main obstacle to an efficient SAO implementation comes from the highly heterogeneous data flows of SC and PD. The SC for each EO or BO classifier comprises many simple iterations. On the other hand, PD involves significantly less iterations (56 or less for each CTU) with each of them being much more complex.

The system throughput (TP) can be regarded as the product of clock frequency ($freq$) and parallelism (N) in the number of samples processed per clock cycle:

$$Throughput = freq.* N \qquad (10)$$

The enhancement of TP can come from the increase of either $freq.$ or N. The serial characteristics of SC and the large number of iterations involved, however, make SC inefficient to be parallelized. As shown in the gray part in Fig. 22 and Fig. 23, the hardware components of these parts have an quadratic growth in area with the increase of N. Detailed quantitative analysis will be given in next subsection. In the meanwhile, the function of SC decides that a short critical path can be achieved, thus a high frequency is preferred. However, a high working frequency is not preferred in PD, because 1) it does not need many clock cycles to perform the limited number of iterations involved for each CTU and 2) each iteration involves the complex computation that results in a long critical path. The big difference in preference to the selection of working frequencies, thus becomes the key challenge for integrating SC and PD efficiently.

3.3.2 The optimal clock frequency of SC

There are many possible combinations of N and $freq$ to support a certain throughput. For instance, N = 16 is used in [42] and [41]. However, there are factors, area and timing, that constrain the choice of N. I list the hardware usage in the crucial modules with N equal to 1, 2, 4, 8, 16 and 32, as shown in III. I explain how area and clock frequency constrain the N and show the optimal frequency as the following.

1) Analysis of area: The modules listed in Table 8 (marked in gray in the Fig. 22) dominate the area consumption when compared with other modules in SC. For these modules, area increases at a growing rate with the increase of N. The increase in area mainly comes from the quadratic growth in quantity of function units (FUs) of EO/BO modules, listed in the second row (EO/BO module) in Table 8. Since BO and EO are similar in architecture, we use EO as an example.

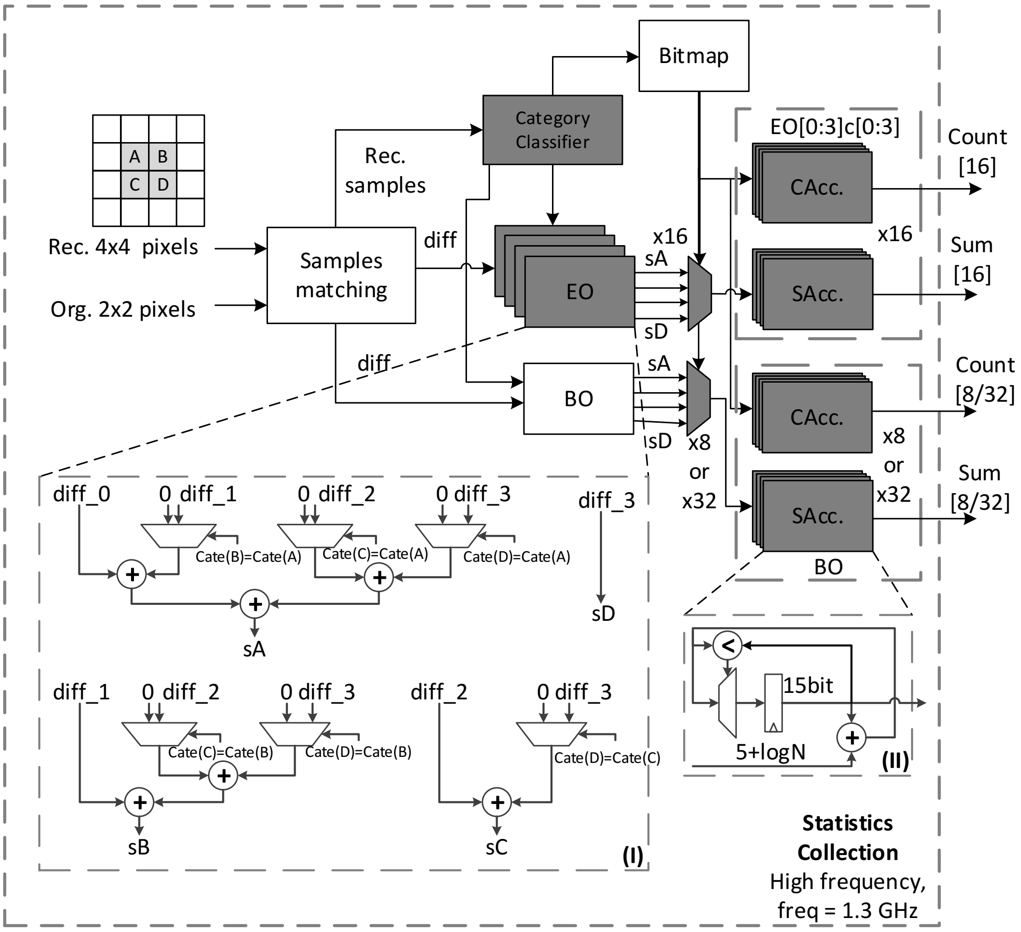


Fig. 22 The architecture of 2×2 statistics collection engine.
The details of dark gray part in Fig. 17. SAcc,CAcc are abbreviations for the accumulators of Sum and Count.

When N = 1, this sample must belong to one of the five categories. By checking the category that this sample belongs to, the corresponding accumulators for Sum and Count (SAcc and CAcc as in Fig. 22) operate. When N = 2, there are two cases for the second sample "B", that B belongs to the same category as A, or not. For the former case, the corresponding SAcc unit increments by the sum of two differences and CAcc increments by two; for the latter case, operation for each sample is the same as the case when N = 1. It could be noticed that the addition of difference of latter samples depends on the result of former ones, because the samples with the same category or band are accumulated together. Considering whether the rest N − 1 samples have the same category with the first sample or not, 2N −1 branches exist and N − 1 adders as well as multiplexers are required. Similarly, when we consider rest N − 2 samples with the second sample, 2N −2 branches exist and N − 2 adders as well as multiplexers are required. Thus we can conclude that N (N−1)/2 adders, multiplexers and comparators are totally required for each EO/BO module at N times of parallelism.

Besides, the number of inputs necessarily multiplexed to each accumulator also grow with the increase of N. As shown in Fig. 22, there are four inputs (sA to sD) for each accumulator when N = 4. Furthermore, the larger data width of each adder also increases the area consumption. The data are shown in the third row (SAccs/CAccs) of Table 8.

2) Analysis of timing: Though the above analysis reveals that the high area efficiency benefits from a smaller N, a smaller N means a higher freq is required to sustain the target TP. However, the maximum frequency is constrained since there is a loop in SAcc and CAcc, as shown in Fig. 22 (II). The path delay, mainly generated by the adder, has a lower bound irrelevant to N. Besides, the path delay in a loop cannot be reduced by pipelining. Thus the achievable clock frequency has an upper bound, or N has a lower bound given a target TP.

3) The optimal clock frequency: To support 8K@120fps, the TP required equals to 7680 ∗ 4320 ∗ 120 ∗ 1.5. From (8) and Fig. 24, we know the required system frequency, 5.2 GHz and 2.6 GHz, are higher than the maximum system frequency (1.5 GHz to 1.6 GHz) when N equals to 1 and 2 respectively. The maximum frequency should always be larger than the frequency required to guarantee a positive slack. Thus N equal to 4 is the optimal choice among the candidates with positive slack, since a smaller N is more area efficient. Thus the corresponding optimal frequency is 1.3 GHz.

Table 8 Relationship between the consumption of hardware resource of crucial modules and the increase of N from 1 to 32. (∗: when applying proposals in section 3.4, this figure is 48, else it is 96.)

Module Name	FUs	Data width	Complexity	# of FU	1	2	4	8	16	32
EO/BO module	Adder	5 bits	O(N2)	2.5N(N-1)	0	5	30	140	600	2480
	MUX	3 bits								
	Comparator	4 bits								
SAccs /CAccs.	Adder	Max (15bits,5bit+logN) Fig. 6 (II)	O(1)	48∗	48					
	Comparator	10~15 bits								
	MUX	2 bits	O(N)	48(N-1)*	0	48	144	336	720	1488
Category Classifier	Adder	3 bits	O(N)	2N	2	4	8	16	32	64
	Comparator	8 bits	O(N)	7N	7	14	28	56	112	224

3.3.3 Other feasible models for parallel SC

1) Multiple sets of accumulators: This model uses N sets of accumulators that work independently and have a final accumulation stage. It gives an O(N) hardware complexity without affecting critical path, but needs (M-1) extra sets of accumulators. The hardware consumption of SAccs/CAccs in Table 8 increases to O(M) from O(1). The experiment result shows Accs. of one category occupy 0.55K gates@650MHz. Thus, at least extra 13K gates are required, even with only 2 sets of accumulators. The proposed model is thus more efficient than the M-set accumulator model as long as N is less than 32.

2) Serial: This model uses 1 set of registers but N sets of multiplexers and adders connected in series. It gives an O(KN) hardware complexity, where K = 48 (4 EO

pattern * 4 categories/EO pattern + 32 bands). Since each category or band need one string of adders and MUXs connected in series. When N is small, K becomes the main influence on the area. From our analysis, this linear model is worse than our chosen model when N is not greater than 16.

Overall, compared with the above two models, the model with N=4 that we choose is the most area efficient for SC.

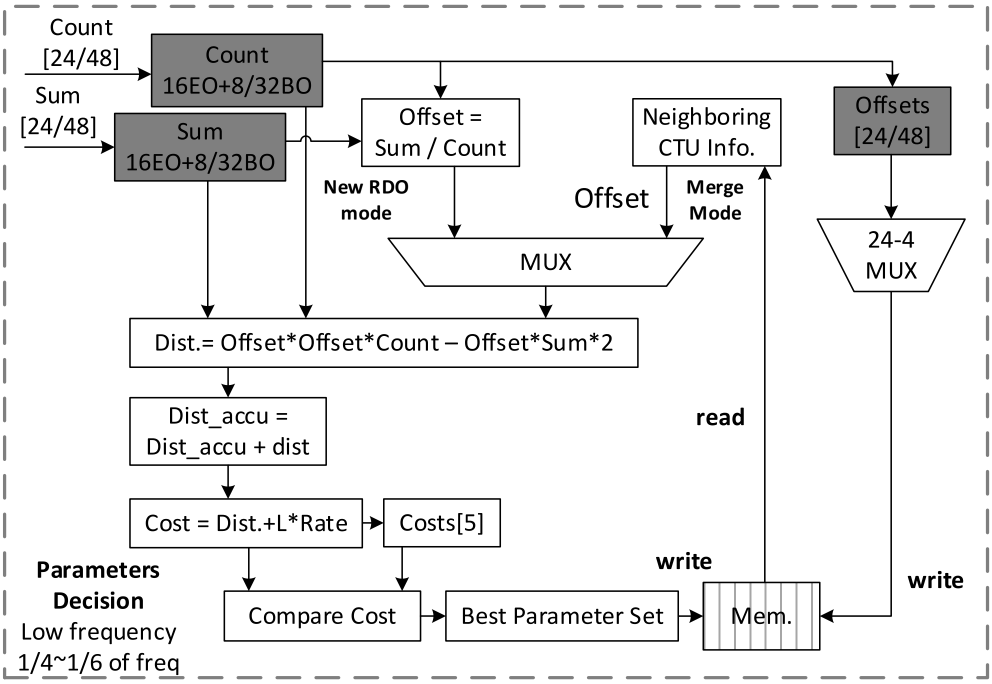


Fig. 23 The proposed architecture of parameter decision Engine.
The details of light gray part in Fig. 17.

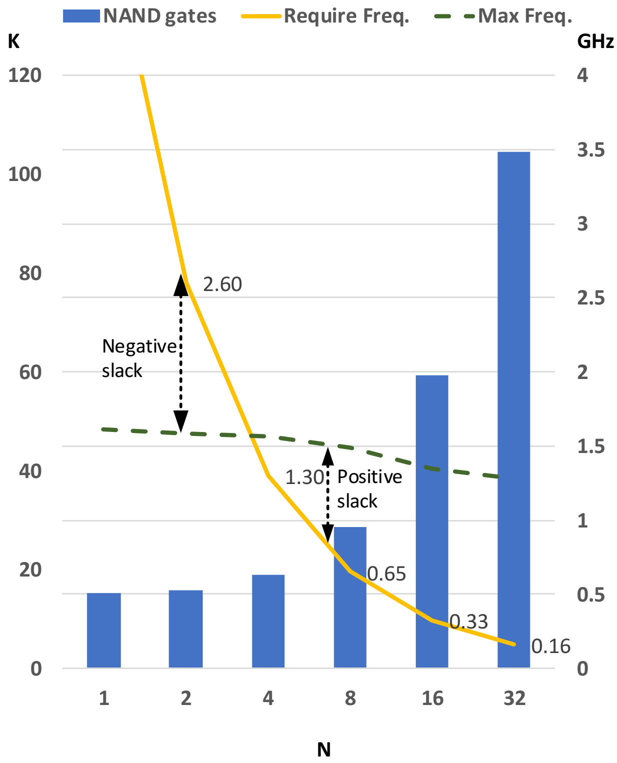


Fig. 24 The relationship among parallelism (N), area and frequency in SC. Under the process of SMIC 40nm, bars represents the approximate number of NAND gates required by the modules listed in Table 8. The dashed line represents the min. frequency required and the solid line represents the max frequency could be achieved with various N.

3.3.4 The optimal frequency in PD

The best parameter sets for each CTB are decided among 4 EO patterns (4 categories in each), 32 bands, and 2 merge categories (4 offsets to be evaluated in each category). Intuitively, it generates a cost for an EO category or a BO band per cycle, and a cost for each merge candidate every 4 cycles. This process is pipelined in three stages. Totally it takes 16 + 32 + 4 + 4 + 4 = 60 cycles for processing the PD of each CTB. Since the result (sum, count) of SC stored in registers are used by PD, the clock frequency of PD should be above a lower bound as the following equation:

$$Freq_{PD} \geq NC_{PD} * N_{CTB} * N_{frame} \quad (11)$$

NC_{PD} is the numbers of cycles to finish PD, N_{CTB} is the numbers of CTB in a frame and N_{frame} is the number of frames encoded in a second.

The $Freq_{PD}$ is enough to support the required throughput with 3 pipeline stages when it is only 1/6 or 1/4 of $Freq_{SC}$ (1.3GHz). Their area are 21K and 25K, respectively. When $Freq_{PD}$ is 1/2 of q_{SC} , 5 pipeline stages are required. Its area is 30.4K gates, with the area overhead being 45%. It's very difficult to increase the frequency of PD to further match the frequency of SC, because the calculation of Offset, Dist. and Cost in PD, as shown in Fig. 23, consists of complicated multiplexing, and multiplication, which is challenging for the deep pipeline.

3.3.5 Proposed architecture

Based on the analysis above, we propose a dual-clock architecture, where a high speed clock drives SC and a low speed clock drives PD, so that the features of each part could be exploited. The frequency of the high-speed clock is 1.3 GHz and the frequency of the low-speed clock is 1/4 or 1/6 of $clk_{high-speed}$. Both of them are derived by clk_{base}. The relationship of frequency between them is shown below.

$$clk_{high-speed} = clk_{base} \tag{12}$$

$$clk_{low-speed} = \frac{clk_{base}}{M} = \frac{clk_{base}}{floor(NC_{SC}/NC_{PD})} \tag{13}$$

In our work, M is four without CRS (to be presented in Section 3.4) or six with CRS.

For SC, it takes 905, 240 and 240 high-speed clock cycles to process a Luma, Cb and Cr CTB in serial, respectively. NC_{SC} is equal to 240, since the minimum number of cycles used in SC is decided by the Cb/Cr channel. For PD, it takes 60 low- speed clock cycles to find the best parameter candidates for each channel. The resulting M is 4. With CRS, M increases to 6, since the number of clock cycles for finding the best set of candidates (NC_{PD}) decreases from 60 to 36 with the number of bands decreased from 32 to 8.

The processing schedule for SC and PD is shown in Fig. 25. SC and PD are processed in pipeline stage s1 and s2, in two different clock domains respectively. The data in s1 are kept unchanged for at least M (4 or 6) cycles, so that the data in s1 could be caught by the rising edge of low-speed clock, and be transferred to s2 before they are updated for another new CTB in s1 during this period.

It is noted that two clocks have a frequency relationship of a dividends M and are derived by clk_{base}. The rising edges of each clock are periodically aligned in delta time, making it unnecessary to have the extra data synchronization. Compared to using two completely independent clocks, it eliminates the hardware expense for an additional phase-locked loops (PLL).

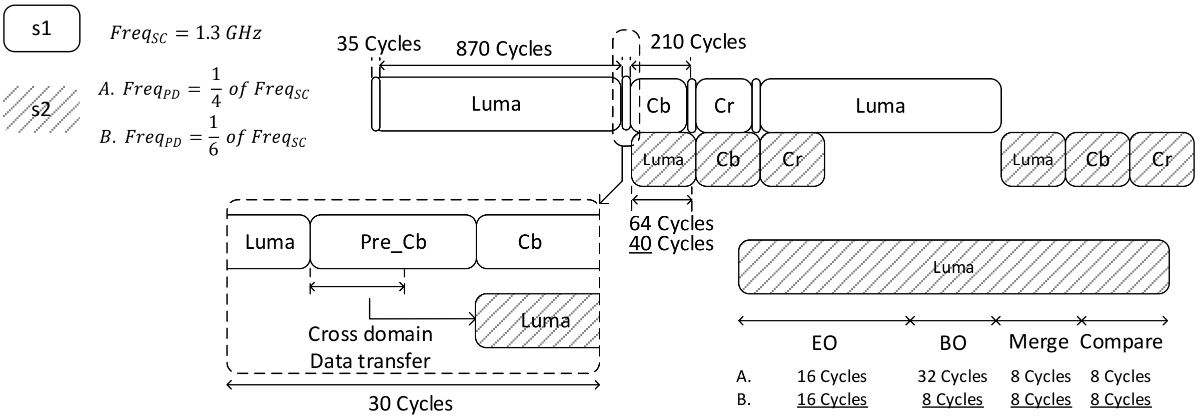


Fig. 25 The schedule of CTB processed in pipeline in two clock domains.

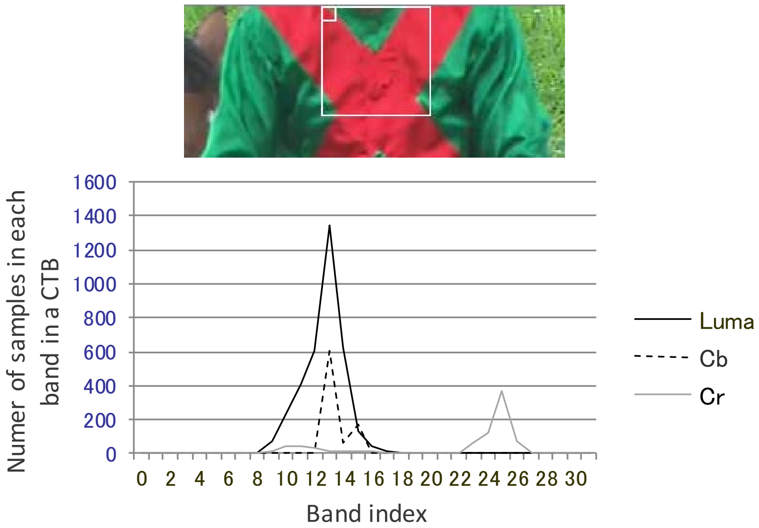


Fig. 26 The number of samples distributed in each band within a CTB.
(E.g, A normal CTB of video sequence of Racehorse 832×480)

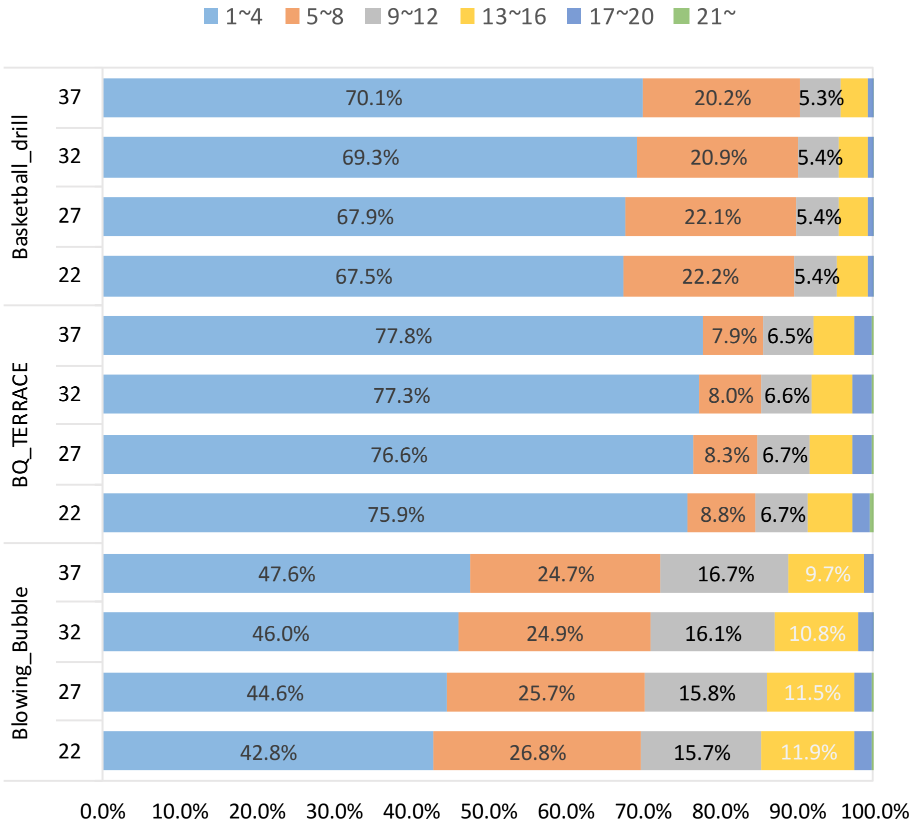


Fig. 27 The distribution of number of bands in a CTB in various sequences.

Video sequences: BQTerrace, BasketballDrill and BlowingBubbles. They are evaluated with Quantization Parameter (QP) of 22, 27, 32, 37 and have the maximum, medium and minimum BD-rate degradation, respectively, as shown in the 3rd column of Table 10.

3.4 Algorithm-Architecture Co-optimization

3.4.1 Coarse Range Selection

The exhaustive search among 32 bands to find the best bands consumes two-third (32/48) of hardware resources of the design, while the hardware for EO dominates the rest one-third.

If we could design a hardware friendly scheme that coarsely select the sample value ranges in each CTB to reduce the search range from 32 bands to a small number before SC, the overall resource for collecting BO statistics can be decreased. The selected range would better cover as many samples within a CTU as possible, so that the band characteristics of a CTU could be mainly preserved. As mentioned in Section 3.1, a BO pre-decision was proposed in [38]. The BO pre-decision scheme searches among all 32 bands to find the best one, which is software oriented and aims at speeding up the band decision process. However, it is the impracticable for hardware implementation since the limitation in SRAM bandwidth causes that lots of cycles are taken to fetch the samples in a CTU for pre-decision.

From Fig. 26 we observe that most of the sample values are distributed in several bands, and distribution of the chroma samples are even more concentrated. I further collect the statistics of the number of bands used in each CTB in video sequences, so as to know how large a range of bands is enough to efficiently classify most of samples in a CTB. The distribution of number of bands used in a CTB is shown in Fig. 27. Results show that 82% of CTBs have at least 90% of their samples concentrated in no more than 8 bands. The distribution is more concentrated when the video sequence becomes larger, since each CTB tends to contain less textures. Such results indicate that the use of 8 bands to collect the statistics could guarantee most of samples to be classified.

I thus propose the CRS for BO based on most likely band estimation. It reduces both the searching space for the best bands and the relevant hardware resource for BO from 32 to 8. Before the start of statistics collection, we define a coarse selection stage, which contains 16 cycles for a CTB. During this stage the system makes an estimation on the bands distribution and finds the center of distribution. In each cycle of this stage, a window of 2×2 reconstructed samples scans 16 locations evenly distributed in a CTB. The average of the samples in the window are calculated and accumulated. In the last cycle of the pre-estimation stage, a final average value is calculated. The band belonging to this value is regarded as the one in the center of sample distribution within the current CTB. The left 3 bands and right 4 bands of it are considered as the reduced 8 band candidates, as shown in Fig. 28. The four consecutive bands with minimum cost from them will be selected in PD. For the pixels outside the 8 ranges, the SC does not collect the statistics of them, which could reduce the coding efficiency. But the reduction is very limited, with average 0.2% BD-rate increase in each configuration.

The experiment result shows that this proposed technique has the top-1 prediction (the proposed best band is the same as the best one from original HM) rate is about 60%, and the top-3 prediction rate is about 80%. More results about the performance of this proposed scheme evaluated in BD-Rate are shown in next section.

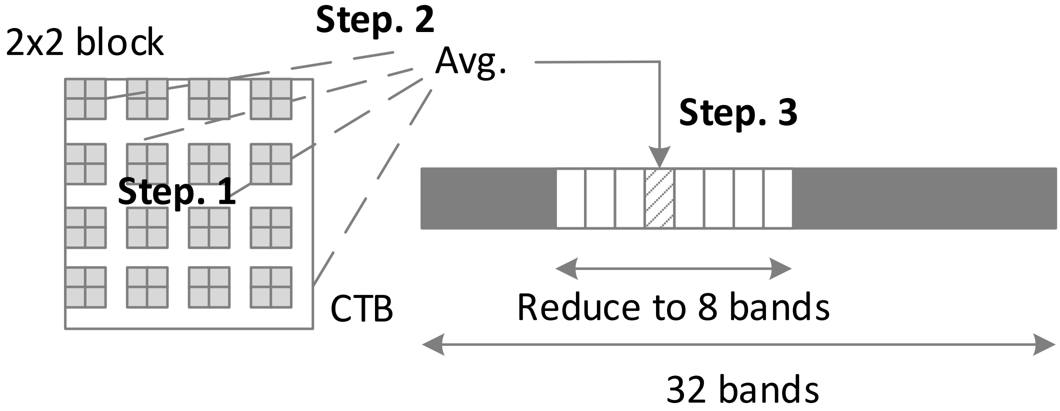


Fig. 28 The process of reduced BO candidates searching.
Step 1 : Scan the 16 evenly distributed windows (2×2 reconstructed samples) one bye one, calculate the average value of the samples in the window, and accumulate the average value.
Step 2 : Estimate the average value within the current CTB. $\text{Avg.} = (\text{Sum} + 8) \gg 4$.
Step 3 : Candidate bands range from $(\text{Avg.} \gg 3) - 3$ to $(\text{Avg.} \gg 3) + 4$.

3.4.2 Accumulator bit width reduction (ABR)

As shown in Fig. 22, there are 24 or 48 SAccs and CAccs (with or without CRS) in SC. Theoretically, each CAcc could increment to the maximum value, 4096 (64×64) for each CTB. Since a sample is classified into only one of the bands/categories, however, the classification is mutual exclusive with each other. In most cases, the final value of CAcc is about several hundreds.

I thus propose to replace the maximum value of CAcc by a smaller threshold, so as to reduce the bit width of CAcc and SAcc. Once the value in CAcc reaches a threshold, the accumulations in CAcc and the corresponding SAcc are terminated. The data width reduction of function units in CAcc and SAcc depends on the threshold.

I compare the effect of BD-rate reduction among three thresholds, 1024, 2048 and 4096. The experiment result shows that there's no observable coding efficiency loss for any them. In fact, we found a threshold of 1024 could still preserve 97% of statistics. The statistics loss has little influence on the coding efficiency.

By setting a threshold of 1024, we could reduce the data width in each CAcc as well as SAcc, contributing area reduction by 5K gates, 10% of the entire area.

Table 9 The BD rate comparison of algorithms under LowDelay_Main_P (LDP) configuration (Anchor : HM-16.0, SAO off, CTB:64×64)

	Video Sequences in common test	**SAO_ON**	**ICIP14 [25]**	**Proposed without CRS or ABR**	**Proposed with CRS or ABR**
Class A 4Kx2K	Traffic	-8.1%	-8.2%	-8.2%	-8.2%
	PeopleOnStreet	-6.0%	-6.5%	-6.4%	-6.4%
	Nebuta	-7.6%	-8.8%	-8.5%	-8.4%
	SteamLocomotive	-16.0%	-17.6%	-17.6%	-17.7%
Class B 1080p	Kimono	-7.0%	-7.8%	-7.7%	-7.7%
	ParkScene	-8.1%	-8.0%	-8.0%	-8.2%
	Cactus	-11.2%	-11.6%	-11.8%	-11.6%
	BasketballDrive	-8.4%	-8.6%	-8.5%	-8.8%
	BQTerrace	-17.1%	-18.2%	-18.2%	-18.1%
Class C WVGA	BasketballDrill	-8.6%	-9.1%	-9.2%	-8.3%
	BQMall	-8.1%	-8.2%	-8.3%	-8.4%
	PartyScene	-4.9%	-4.9%	-5.0%	-5.0%
	RaceHorses	-8.8%	-9.0%	-9.0%	-9.0%
Class D WQVGA	BasketballPass	-4.6%	-4.8%	-4.6%	-4.7%
	BQSquare	-4.4%	-4.4%	-4.4%	-3.7%
	BlowingBubbles	-4.2%	-4.3%	-4.3%	-4.3%
	RaceHorses	-6.1%	-6.1%	-6.0%	-6.2%
Class E 720p	FourPeople	-9.2%	-9.6%	-9.6%	-9.5%
	Johnny	-12.3%	-12.4%	-12.4%	-13.1%
	KristenAndSara	-11.2%	-12.2%	-12.1%	-12.0%
Class F	BasketballDrillText	-9.2%	-9.5%	-9.6%	-8.3%
	ChinaSpeed	-9.7%	-10.9%	-10.8%	-8.0%
	SlideEditing	-4.2%	-2.5%	-2.3%	-1.3%
	SlideShow	-7.1%	-7.8%	-8.0%	-5.5%
Class Summary	Class A	-9.5%	-10.3%	-10.2%	-10.2%
	Class B	-10.4%	-10.8%	-10.8%	-10.9%
	Class C	-7.6%	-7.8%	-7.9%	-7.7%
	Class D	-4.8%	-4.9%	-4.8%	-4.7%
	Class E	-10.9%	-11.4%	-11.4%	-11.5%
	Class F	-7.6%	-7.7%	-7.7%	-5.8%
	All	**-8.4%**	**-8.8%**	**-8.4%**	**-8.4%**

Table 10 The BD rate comparison of proposed algorithm with CRS and ABR under different configuration, All Intra(AI),Random Access(RA), and LowDelay main(LD) (Anchor : HM-16.0, SAO on, CTB :64×64)

	Video Sequences in common test	**AI**	**RA**	**LD**
Class A **4Kx2K**	Traffic	0.2%	0.2%	0.2%
	PeopleOnStreet	0.2%	-0.1%	-0.2%
	Nebuta	0.2%	-1.0%	-1.2%
	SteamLocomotive	-0.2%	-0.4%	-0.8%
Class B **1080p**	Kimono	0.2%	0.1%	0.0%
	ParkScene	0.2%	0.2%	0.0%
	Cactus	0.3%	0.1%	0.1%
	BasketballDrive	0.4%	0.1%	-0.1%
	BQTerrace	-0.1%	-0.1%	-0.4%
Class C **WVGA**	BasketballDrill	0.7%	0.6%	0.6%
	BQMall	0.3%	0.1%	-0.0%
	PartyScene	0.1%	0.1%	-0.0%
	RaceHorses	0.2%	0.0%	-0.2%
Class D **WQVGA**	BasketballPass	0.4%	0.1%	-0.0%
	BQSquare	0.1%	0.6%	-0.1%
	BlowingBubbles	0.1%	0.1%	-0.1%
	RaceHorses	0.2%	0.1%	-0.2%
Class E **720p**	FourPeople	0.2%	0.3%	0.0%
	Johnny	0.5%	0.5%	0.2%
	KristenAndSara	0.4%	0.5%	0.0%
Class F	BasketballDrillText	0.6%	0.7%	1.8%
	ChinaSpeed	0.9%	1.3%	1.8%
	SlideEditing	0.5%	1.5%	2.7%
	SlideShow	0.6%	1.0%	1.7%
Class **Summary**	Class A	0.1%	-0.3%	-0.5%
	Class B	0.2%	0.1%	-0.1%
	Class C	0.3%	0.2%	0.1%
	Class D	0.2%	0.2%	0.0%
	Class E	0.4%	0.4%	0.1%
	Class F	0.7%	1.1%	2.0%
	All	**0.3%**	**0.3%**	**0.3%**

Table 11 Comparison of synthesis result with the previous works

	This work		ICIP'14 [25]	ISCAS [24]
Process	40nm		65nm	28nm
Area (gates)	51K		156.3K	300K
SRAM	1.14KB		1.08KB	N/A
TP (pixels/s) for encoding	4320p, 120fps		4320p, 120fps	2160p, 60fps
Cycles to finish 64x64 CTB:SC	905		384	1600
Cycles to finish 64x64 CTB:PD	40		64	N/A
Clock Freq. (MHz)	SC: 1300	PD: 217	SC & PD: 378	SC & PD: 200
Norm. TP (samples/(gates*s))	117.1K		38.2K	2.8K

Table 12 Comparison of the proposals with and without CRS and ABR

		Proposed without CRS or ABR	Proposed with CRS or ABR
Area (gates)	**SC modules**	43K	30K
	PD modules	25K	21K
	Total	68K	51K
Cycles to finish 64x64 CTB	**SC modules**	870	905
	PD modules	64	40
SRAM		1.14KB	
TP (pixels/s) for encoding		4320p, 120fps	

3.5 Experimental Results

The proposed AD design has been implemented on RTL in SystemVerilog. Logic synthesis and physical design have been conducted with Synopsys Design Compiler and Cadence SoC Encounter, respectively, in SMIC 40nm CMOS standard cell library. For verification, input data and expected outputs were generated from HM 16.0 software model as stimulus and reference for the hardware design. The layout is shown in Fig. 29. The high speed clock domain can work under the required frequency of 1.3 GHz. I evaluate the power consumption for the video sequence of BasketballDrill with QP=37 and under the low delay, main, P slices only (LDP) configuration. The power of our design is 48 mW when high speed clock equals to 1.3 GHz.

To evaluate the coding efficiency of the proposed SAO, two groups of tests are conducted over the common test condition [44]. The first group of tests evaluate the following five algorithms, including the HM-16.0 default setting with SAO turned off (anchor) and turned on, algorithm in [42], the proposed algorithm without and with CRS as well as ABR. This group is evaluated under LDP configuration. Since the effect of SAO is the most obvious [22] under this condition. The result is shown in Table 10. The second group of tests evaluate the following two algorithms, the HM-16.0 default setting with SAO turned on (anchor) and the proposed algorithm with CRS and ABR. This group is evaluated under three configurations, All Intra (AI), Random Access (RA) and low delay (LD). The result is shown in Table 11.

Comparing with the proposed design with [42] in Table 12 and Table 10, we know that our proposed design achieves a reduction by 69% from [42] with no coding efficiency loss in BD-Rate. Compared with the result the anchor in Table 11 under various configuration, our proposed algorithm has 0.3% coding efficiency loss in BD-Rate. This is achieved by the following techniques.

First, the parallelism of SC is reduced from 16 in [42] to 4 in this work, under the requirement of meeting throughput of 8K@120fps. As is illustrated in Section 3.3, the cost of increasing the paralleling factor N is large in SC of SAO. The paralleling factor of 4 is much more area efficient than the factor of 16. Thus, our work reduce the circuit area by 67%, from 156.3K to 68K, based on the same specification and algorithm setting.

Second, we found that the CRS for BO nearly has no observable loss in coding efficiency in terms of BD-rate. Both the data shown in Fig. 27 and the top-3 prediction rate of 80% reveal that the range selected by our scheme cover most of samples in a CTU. The cost of best three bands are so close that even though the best is missed by our scheme, there are 2nd and 3rd best to compensate the statistics loss. The differences on BD-Rate performance among the best, 2nd and 3rd best bands are small. The proposed algorithm-architecture co-optimizations can further reduce the circuit area by 25%, from 68K to 51K, as shown in Table 9. The reduction in area is contributed by the following aspects. 1) CRS decrease the overall SAcc and CAcc in SC decrease by 50%, from 48 to 24. The registers used for storing offsets also decrease by 50%, from 48×3 to 24×3. It achieves a 13 K area reduction, with an area overhead of 2K in the pre-decision step. 2) CRS also reduces the number of cycles for PD from 60 cycles to 36 cycles, so that a looser time constraint for PD, increased from 2.8 ns to 4.8 ns, further reduces the circuit area by 3K. 3) ABR helps to reduce the data width in each CAcc and SAcc, contributing 5K gates reduction with BD-rate increase of 0.1% and 0.2% in LD and LDP configuration respectively.

When synthesized in the same Fujitsu e-Shuttle 65nm process as in [42], our design has a logic gate count of 59.6K, with 35.5K and 24.1K for SC and PD, respectively. I give a brief comparison on power consumption by analyzing the area, frequency and switching factor. The area-frequency product of our work is 35.5K*1.3G + 24.1K*0.217G = 51.3P, 20% lower than that of [42]: 156K*0.4G = 64.4P. Moreover, the high- parallelism SC of [42] involves the updating of more accumulator registers per clock cycle, resulting in a higher switching factor. As a result, our design is more energy efficient than [42] with reduction in both area-frequency product and switching factor.

I have also implemented an N=8, 650MHz version to study the area and power consumption, which turns out to be 60.5K and 32mW, respectively. The N=8 version therefore is 20% larger in area, but 33% more efficient in power. The latter is mainly from the fact that a looser timing constraint compared to the N=4, 1.3GHz configuration now allows the synthesizer to use slower (and therefore smaller and less power consuming) logic cells. Moreover, we roughly implemented and estimated that the N=16 version is twice in area compared with the N=8 version, where an even looser timing constraint does not influence results much. Overall, the design of N larger than 8 does not show higher efficiency in energy despite being significantly larger in area. The N=4 version is more efficient in area, while the N=8 version is more efficient in power. Both can be taken into considerations for applications.

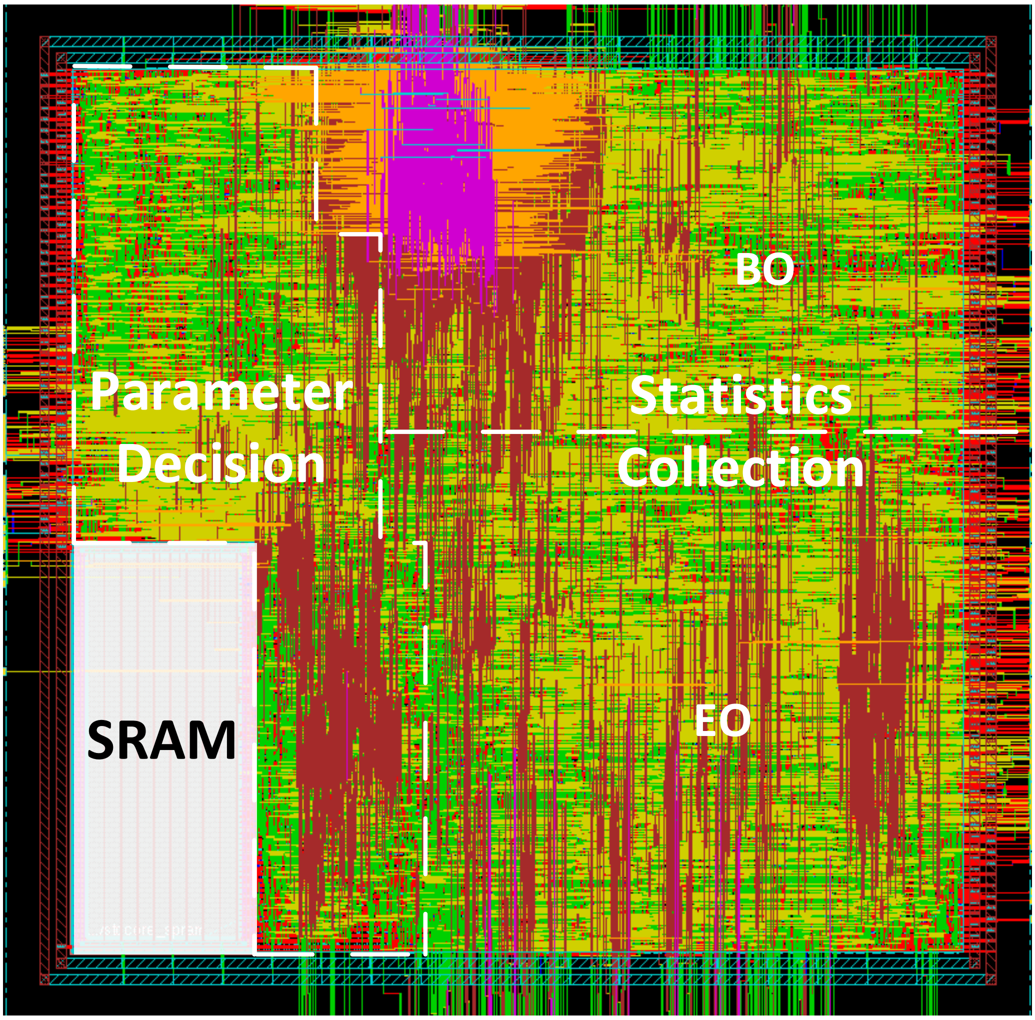


Fig. 29 Layout of the proposed SAO architecture.
A core size of 1.73 mm^2 and chip-area utilization of 73%. Before the layout, the delay of the critical path is 0.66 ns, and after the layout, it is 0.76 ns. The location of PD and SC is marked, while the SRAMs for line buffer are marked in white.

3.6 Summary

SAO is a new in-loop filter in H.265 video coding standard. Many researches have been trying to improve its performance and area efficiency of hardware design. This paper presents an efficient VLSI design of SAO estimation in H.265. I first introduce SAO and analyze its data flow. Then we proposed the dual-clock architecture to address the heterogeneous data flows of SC and PD, by separately driving SC and PD at a high-speed clock and a low-speed clock, respectively. Two clock frequencies with a relationship of dividends M eliminates the extra hardware and implementation expense. Moreover, the algorithm-architecture co-optimizations, CRS and ABR further reduce the circuit area without observable loss in coding efficiency. The proposed architecture occupies 51 K logic gates. With a high-speed clock of 1.3 GHz and a low-speed clock of 217 MHz, 8K@120fps SAO real-time encoding can be achieved. The demerit of the proposal is that the local heat problem would exist due to the high frequency clock.

4. Algorithm and VLSI architecture of intra prediction in Compressed Sensing using reduced measurements

4.1 Introduction

The Internet of Things (IoT) or Machine-to-Machine (M2M) network has been widely discussed in recent years and is regarded as the next wave of the information technology revolution [46]. Sensors, as the troops of IoT, are the on-the- ground pieces of hardware that monitor processes, collect and transmit data. Among the various types of sensors, image sensors are those collecting and processing the largest amount of data. In M2M networks, massive deployments of wireless camera systems (image sensor nodes) are required. Since they are highly battery-constrained devices, low power consumption is a fundamental concern. Conventionally, a wireless camera system comprises three main components: the CMOS image sensor (CIS), the compressor, and the transmitter. Images are acquired by the CIS, which converts the illumination of light into a digital signal pixel-by-pixel. The digital signals are compressed by the compression unit using encoding algorithms, such as JPEG [47], H.264/MPEG- 4 [48], DVC [49], and H.265 [7], before they are transferred to the channel. This is the traditional procedure: Capture → Compress →Transmit, as shown in Fig. 30 (a).

With the advent of a recently proposed sampling theory, Compressed Sensing (CS) [51], the capturing and compression can be performed in CIS simultaneously. Such image sensors are called CS-based CIS (CS-CIS). In CS-CIS, an image is acquired by sampling a significantly reduced number of measurements (the linear combination of pixels), in- stead of by sampling every pixel, and therefore this technique could reduce the throughput of Analog-to-Digital (A/D) conversion, as shown in Fig. 30 (b). This reduction in throughput has the potential to reduce power consumption and increase the frame rate [52], which has been shown in the recently emerging CS-CIS systems [53][54][55]. Since the output of CS- CIS are measurements instead of pixels, however, the spatially adjacent correlation in the pixel-domain is corrupted during the generation of measurements. Hence, traditional intra coding methods cannot be applied to CS-CIS. A direct reconstruction of pixels from the measurements prior to the encoder does not work either, owing to the high complexity and power consumption of reconstruction [51]. Therefore, a direct measurement-domain compression method with low complexity is desirable. It also ought to be compatible with the CS-CIS, which uses a binary or ternary random matrix as measurement matrix.

Several previous works have studied exploiting the spatial redundancy in the measurement-domain for image compression. In [56], measurements in previous blocks were directly subtracted and used for prediction. Nevertheless, it only partly utilizes the horizontal correlation. In [57] [58], the intra prediction occurs by the measurement-wise subtraction from the neighboring measurements, similar to pixel-wise subtraction inter prediction. In these works, however, the measurements for prediction contain irrelevant information, such as the nonadjacent pixels, so that the pre- diction precision decreases. In [59], a local structural measurement matrix providing more precise prediction is proposed for the measurement-domain prediction by extracting the local features within a block. However, it has high computational complexity for a brute-force search among all the local predictor candidates, and it requires a floating point measurement matrix that cannot be applied to the image sensor. Another issue in [57] [58] [59] is that they require all the measurements of a block for prediction. It requires a large memory bandwidth to fetch / load the data, as well as a large memory storage used for line buffer to store all the measurements of neighboring blocks, which would be a problem for a power-limited and a storage-limited wireless camera system. Overall, the previous works were not designed oriented to CS-CIS, which is required to generate simple (binary or ternary) coefficients for image compressive sampling.

In this paper, which is an extension of our previous work [60], we propose a measurement-domain-based intra prediction coding framework as well as its VLSI implementation, containing the following features:

1) A higher compression ratio achieved: By structuring two rows in the random binary measurement matrix, the average values of the neighboring block's row and

column are embedded into two measurements, such that they could be extracted during the intra coding process to perform more precise prediction.

2) Low-complexity framework compatible with CS-CIS: It is based on compressively sensed images that take the measurement rather than pixels as input to the encoder, such that the recovery of pixels from measurements is avoided. Moreover, two artificially structured rows retain the binary property of the random matrix, which is crucial for the compressive sampling in CS-CIS.

3) Low hardware cost: The proposed prediction algorithm makes the intra prediction hardware-friendly. The matrix multiplication could simply be substituted by the shared adder and shift operation. Furthermore, it reduces the size of neighboring information fetched / loaded for pre- diction, which significantly reduces the memory band- width and storage for the line buffer.

Experimental results demonstrate that the VLSI architecture of our proposed framework is 9.1K gates in area and includes 12 KB dual-port SRAM memory. It could support the 4320p@240fps real-time encoding. Compared to the direct output of CS-based sensors, our proposed framework could compress the measurements and increase coding efficiency with 34.9% BD-rate reduction. Compared to the previous work [57], this work increases coding efficiency with 7.7% BD-rate reduction and saves 83% size of memory bandwidth and storage for line buffer and left neighboring buffer. It can significantly reduce both the energy consumption and bandwidth in communication.

The rest of this paper is organized as follows. Section 4.2 gives an introduction to compressed sensing. Section 4.3 presents the proposed framework of measurement-domain intra prediction. Section 4.4 presents the VLSI implementation of the proposed framework. Section 4.5 shows the implementation results and gives some analysis on the performance of the proposed algorithm and architecture, followed by the conclusion in Section 4.6.

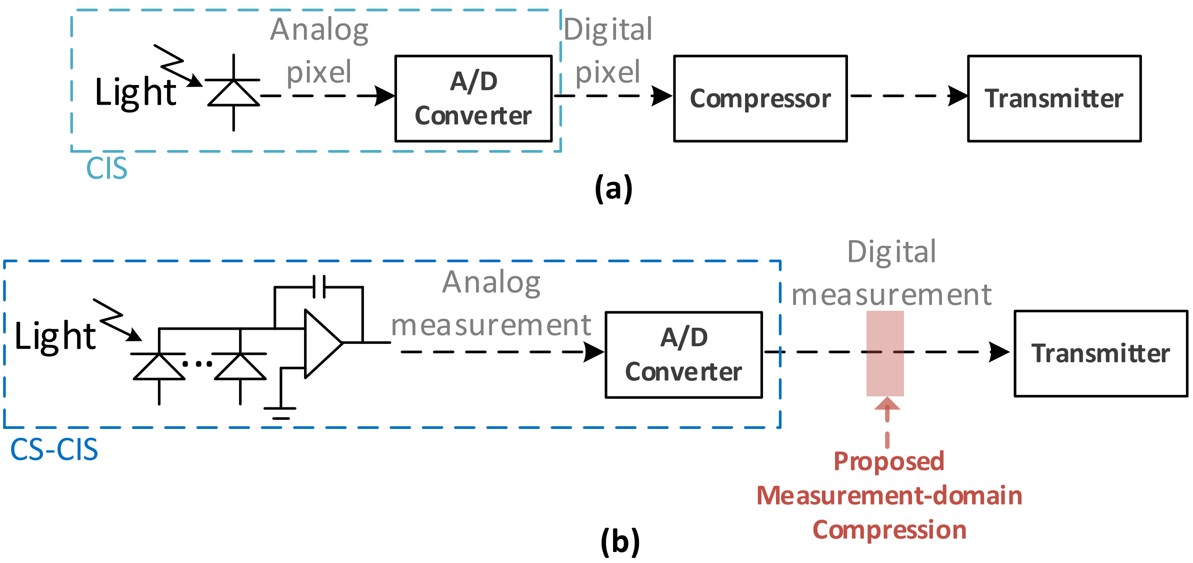


Fig. 30 Data flow of imaging

(a) Data flow of traditional imaging: Capture →Compress →Transmit. (b) Data flow of compressive imaging: Capture →Compress →Transmit. (The red part shows where this work is in the data flow).

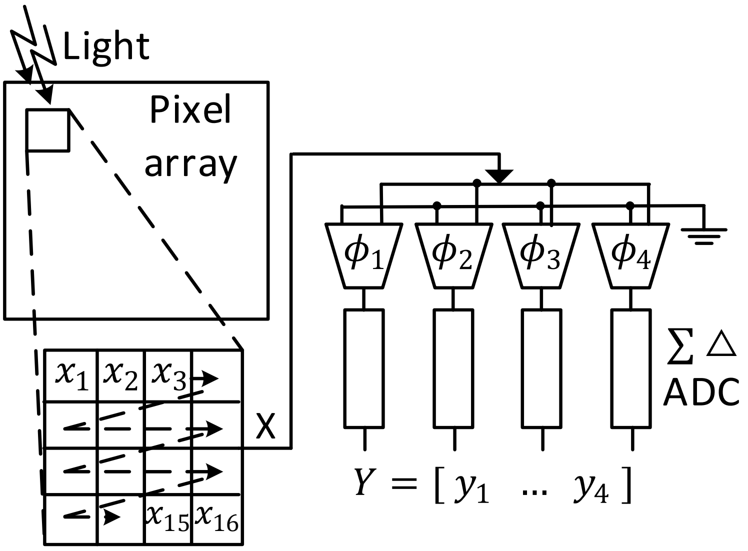


Fig. 31 Brief architecture of CS-CIS.

e.g. As the architecture in [9], each block is 4×4, including 16 pixels and the sampling rate is 1/4. Φ1 to Φ4 are four rows of the matrix Φ.

4.2 Compressed Sensing (CS)

4.2.1 Concept of CS:

The CS theory [51] asserts that only a few measurements are enough to recover the signals, as long as the signals are sparse in some transform domain. Suppose the image signal $\mathrm{X} = [\mathrm{x}_1 \ldots \mathrm{x}_n]^T$can also be represented in the transform domain Ψ, as

$$\mathrm{X} = \Psi\mathrm{S} \tag{14}$$

where $\mathrm{S} = [\mathrm{s}_1 \ldots \mathrm{s}_n]^T$ is the signal represented in Ψ transform domain and Ψ is an n × n transform matrix. The signal X is said to be k-sparse if it has only k non-zero coefficients.

I would like to recover signals $\mathrm{X} = [\mathrm{x}_1 \ldots \mathrm{x}_n]^T$ from $\mathrm{m} \ll \mathrm{n}$ linear and non-adaptive measurements $\mathrm{Y} = [\mathrm{y}_1 \ldots \mathrm{y}_m]^T$, which are taken from the random projection as

$$\mathrm{Y} = \Phi\mathrm{X} \tag{15}$$

where Φ is an m × n measurement matrix. I know that the system is under-determined since m < n. The CS theory asserts that the signal S' can be recovered with high probability using only $\mathrm{m} = \mathrm{cklog}(\frac{\mathrm{n}}{\mathrm{k}})$ measurements for some constant C, by solving the L1-norm minimization problem (3)

$$\min \|S'\|_1 \ \mathrm{s.t}\, \mathrm{Y} = \Theta\mathrm{S}' \tag{16}$$

where $\Theta = \Phi\Psi$ and the measurement matrix Φ must be in- coherent with transform matrix Ψ to preserve the Restricted isometry property (RIP) [51]. The CS theory shows that Φ can even be a random 1/−1 or 0/1 matrix, while Ψ could be a discrete cosine transform (DCT), discrete wavelet transform (DWT), contourlet transform and so forth.

The problem (16) can be solved by basis pursuit [51]. To a noise environment, (16) can be extended to $\mathrm{Y}' = \Theta\mathrm{S}' + \mathrm{Z}$, where Z represents the noise. It could be solved by basis pursuit denoising [51]. After the recovery of S', the signal X' can thus be calculated by (14).

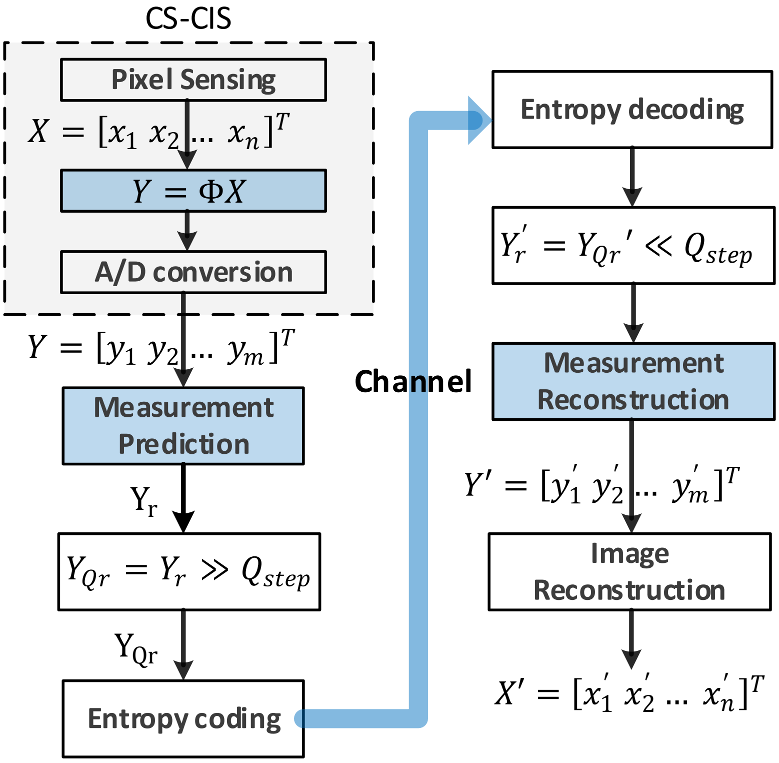

CS-CIS
Pixel Sensing
$X = [x_1\ x_2 \dots x_n]^T$
$Y = \Phi X$
A/D conversion
$Y = [y_1\ y_2 \dots y_m]^T$
Measurement Prediction
Y_r
$Y_{Qr} = Y_r \gg Q_{step}$
Y_{Qr}
Entropy coding
Channel
Entropy decoding
$Y_r' = Y_{Qr}' \ll Q_{step}$
Measurement Reconstruction
$Y' = [y_1'\ y_2' \dots y_m']^T$
Image Reconstruction
$X' = [x_1'\ x_2' \dots x_n']^T$

Fig. 32 Process of CS-based image sensing, encoding, and decoding. The proposed work is indicated by the white box.

4.2.2 Process of CS image sensing:

In the CIS, the A/D conversion is the dominant source of power consumption [54]. The advent of CS theory promises that the recovery can be achieved from the significantly reduced number of captured measurements, hence reducing the A/D conversions for the measurements and their related power consumption.

Fig. 31 shows the principle of CS-CIS. The luminance is sensed by the pixel array. Analog pixel signals are summed up to yield measurements, which are then digitalized by the A/D converters. Note that the generation of measurements is controlled by the elements in the measurement matrix. Therefore, a simple enough binary (0/1 or −1/1) or ternary (−1/0/1) matrix is used in CS-CIS to simplify the measurement calculation so that the complex and energy-consuming analog multiplier could be avoided in the implementation. Moreover, considering the infeasibility and scalability of the image sensor implementation and the complexity of image recovery, the pixel array is divided into blocks to perform the sampling [54] [55].

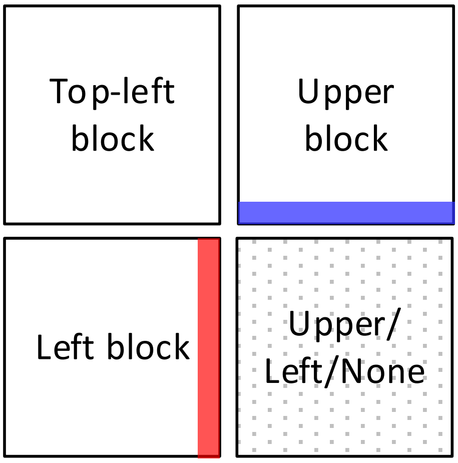


Fig. 33 Predictor candidates
(bottom row of upper block and right-most column of left block.)

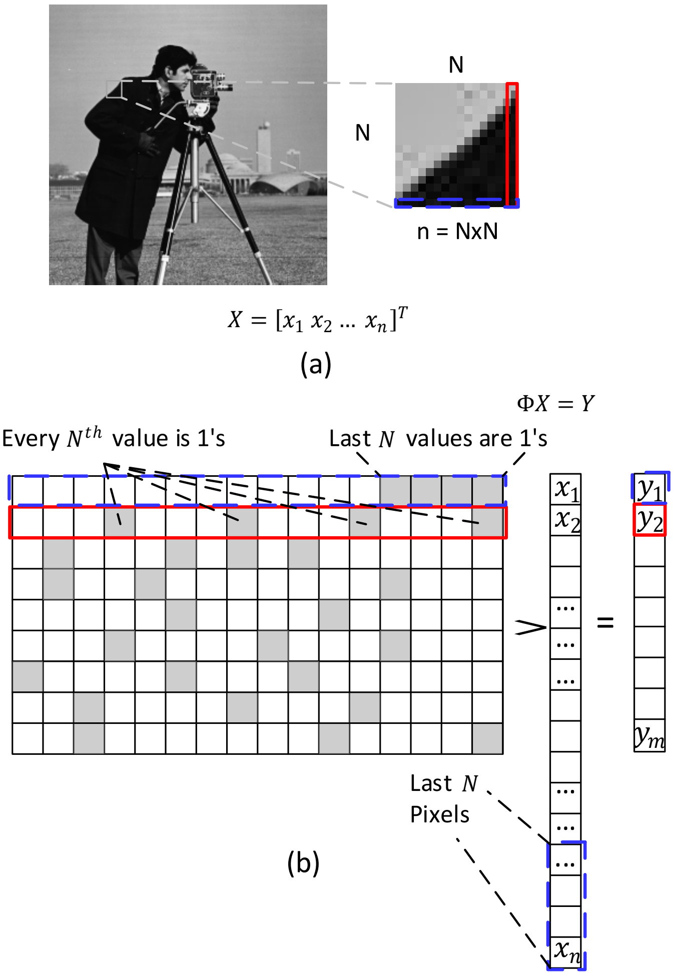


Fig. 34 Proposed artificially structured rows.
(a) N × N block being processed. (b) Mechanism of a structural random binary (0/1) matrix. The last N pixels are summed up by multiplying the 1st row. Every $\mathrm{N^{th}}$ pixel is summed up by multiplying the 2nd row.

4.3 Proposed Coding Framework Based on Measurement-Domain Intra Prediction

4.3.1 Proposed framework

As shown in Fig. 32, for each block: 1) The analog signals of n pixels are acquired by the pixel array inside the sensor, $\mathrm{X} = [\mathrm{x}_1 \ldots \mathrm{x}_n]^T$. The measurements are calculated through (15) in the analog domain and digitalized into $\mathrm{Y} = [\mathrm{y}_1 \ldots \mathrm{y}_m]^T$. 2) Intra prediction is performed on measurements before the quantization, entropy coding, and transmission. 3) The bit stream obtained from the channel is decoded, dequantized, and then reconstructed into measurements. 4) The reconstructed signal X' is recovered from the decoded measurements Y' by solving (16).

In the pixel domain, adjacent pixel values are similar. This property is exploited by the traditional intra prediction, in which the adjacent pixels in neighboring blocks are used as predictors. Pixels with shorter distance tend to be better predictors. In the measurement domain, however, measurements within a block have no similarity with each other, making it difficult to apply the traditional intra prediction. In spatially directional predictive coding (SDPC) [57], the predicted measurements are selected from one of the four designed prediction modes. However, the measurements selected as the predictors are a combination of all the pixels (nearby and far away) within a neighboring block, resulting in a long prediction distance (and therefore low prediction accuracy) on average. Thus, we propose a prediction algorithm by only picking the nearby pixels for prediction to increase the coding efficiency.

Inspired by traditional intra prediction [61], we propose to use the boundary information (bottom row of upper block and the right-most column of left block) as predictor candidates, as shown in Fig. 33. A structural random 0/1 measurement matrix is proposed to extract the boundary information to generate the measurement predictor Y_p , so that $\mathrm{Y}_\mathrm{r} = Y - Y_p$ smaller with a more concentrated value distribution than Y.

4.3.2 Structural measurement matrix

A random 0/1 measurement matrix is used in the CS-CIS [54] [55], because of its hardware friendliness in implementation. I propose to structure the first two rows of the random 0/1 m × n measurement matrix. In the first row, the last N values are set to 1's, while the rest are set to 0's. In the second row, every N^{th} values are set to 1's, while the rest are set to 0's. When the signal X projects to the structural random measurement matrix, the first two measurements have special meaning in the projection. The measurement y_1 represents the sum of pixel values in the bottom row of a block and the measurement y_2 represents the sum of pixel values in the right-most column, as shown in Fig. 34 (b). Though two rows are artificially structured, it could be regarded as one of the random cases. The experimental results show that it could preserve the RIP without affecting the reconstruction of image quality.

4.3.3 Measurement-domain intra prediction

From the first measurement of the upper block $y_{1_{up}}$ and block size N, the average pixel value of the bottom row (BR) in the upper block, $Ave_{BR_{up}} \in R$, can be easily obtained by the shift operation in hardware. Similarly, the average pixel value of the right-most column (RC) of the left block $Ave_{RC_{le}}$ can be obtained from $y_{2_{le}}$. Since the BR in the upper block and the RC in the left block are the most adjacent pixels to the block being processed, they have similar values to this block in the pixel domain. By projecting $Ave_{BR_{up}}$ and $Ave_{RC_{le}}$ to the measurement matrix, the corresponding two measurements generated would be close to the original measurements and thus could be regarded as measurement predictor candidates. The average values $Ave_{BR_{up}}$ and $Ave_{RC_{le}}$ are transformed into measurements Y_{up} and Y_{le} by multiplying the measurement matrix Φ as the follows:

$$X_{up} = \left[ave_{BR_{up}} \ldots ave_{BR_{up}}\right]^T \quad (17)$$

$$Y_{up} = \Phi X_{up} \quad (18)$$

$$X_{le} = \left[ave_{RC_{le}} \ldots ave_{RC_{le}}\right]^T \quad (19)$$

$$Y_{le} = \Phi X_{le} \quad (20)$$

where $X_{up}, X_{le} \in R^{n \times 1}$. It is noted that the calculation of (18) and (20) has a low computational complexity. Since the measurement matrix Φ is known and fixed, it could be achieved by shift operation and addition. The original measurements Y can be treated as the sum of a constant value (e.g. 128) and residuals.

$$Y_C = \Phi[C \ldots C]^T \quad (21)$$

By comparing the sum and difference (SAD) of the original measurements Y and measurements in the three modes, Y_{up}, Y_{le} and Y_C, the measurements with the minimum SAD are chosen as predicted measurements Y_P.

$$p = argmin_{mode} \ SAD(Y, Y_{mode}) \quad (22)$$

$$Y_r = Y - Y_p \quad (23)$$

After the prediction, the residual Y_r is calculated, before being scalar-quantized as follows. The quantized residuals are entropy-coded before being transferred to the channel.

$$Y_{Qr} = Y_r \gg Q_{step} \quad (24)$$

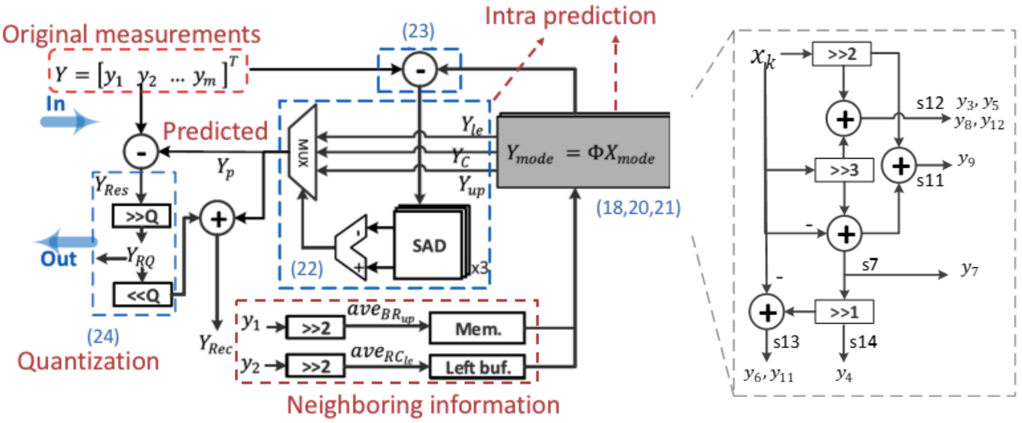


Fig. 35 VLSI architecture for the proposed measurement-domain intra prediction framework.

The dashed boxes are marked by numbers in parentheses, corresponding to the equations in Section 4.3.2. The matrix multiplication is simplified into the shift-add operation in the dashed box on the right.

$$\Phi_4 = \left[\begin{matrix}
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1
\end{matrix}\right]
\begin{matrix} 4 \\ 4 \\ 12 \\ 14 \\ 12 \\ 13 \\ 7 \\ 12 \\ 11 \\ 13 \\ 12 \\ 13 \end{matrix}$$

Fig. 36 Example of random binary matrix of N = 4.

Proposed artificially structured rows are the first two rows in bold. The number of 1’s in each row is shown next to the matrix.

4.4 VLSI implementation of the proposed intra coding framework

The proposed measurement-domain intra prediction framework above is implemented into VLSI architecture, as shown in Fig. 35. The system takes the measurements of block size N = 4 [55] as input, with sample rate (SR) of 0.25, 0.5, and 0.75. Hence, the supported numbers of measurements m as input are 4, 8, and 12, respectively. The output is the quantized residuals Y_{Qr}. A 4 × 4 block is processed every cycle. It is noted that the low complexity of the proposed coding framework allow the hardware be achieved at low cost, as shown in the following two aspects.

First, it could reduce the logic gates used in the calculation, since the multiplication in (18), (20), and (27) could be simplified into the shift and add operation. Taking a block size N = 4 and number of measurements m = 12 as an example, the matrix is shown in Fig. 36. The calculation of the two key measurements y_1, y_2 could be achieved by a shift operation. The rest of the rows contain other constant numbers of 1's, such as 7, 11, 12, 13, and 14. Since they are fixed in a matrix for any CS-CIS and all the elements of X in (17) or (19) are the same, the matrix multiplication in (18) or (20) could be efficiently implemented by shared adders, as in the structure in the right-hand dashed box of Fig. 35. Though the variation of Φ in different CS-CIS requires the shared adders structure above to be designed specifically, the underlying idea of the shift-add operation could be uniformly applied.

Furthermore, the proposed framework could reduce the size of memory bandwidth and storage for the neighboring information. In the previous works, such as [57] [58], all the measurements of the upper block and left block are required to be stored for the prediction. The number of measurements in a block grows quadratically with N and linearly with sample rate (SR), as (25). The number of measurements overall to be stored M grows linearly with the frame size, block size, and sample rate, as (26).

$$M_{Blk} = N^2 \times SR \tag{25}$$

$$\begin{aligned} M &= M_{lineBuf}. + M_{leftBuf.} \\ &= Num_{Blk} \times M_{Blk} + M_{Blk} \\ &= width \times N \times SR + N2 \times SR \end{aligned} \tag{26}$$

The proposed prediction, requiring two measurements for each block ($M' = 2$), could significantly reduce the number of measurements to be stored to M'.

$$M' = M'_{lineBuf.} + M'_{leftBuf.} = \frac{2width}{N} + 2 \tag{27}$$

The comparison is depicted in Fig. 37. As shown in (a), the measurements stored for prediction are inversely proportional to the block size, since the number of block is getting less while only two measurements are required in each block for prediction.

Meanwhile, as shown in (b), the measurements stored grow linearly with the size of the block in [57]. Since the data width of each measurement ranges from 12 bits to 18 bits when block size ranging from 4 to 32, this proposed prediction could significantly reduce the size of memory bandwidth and storage.

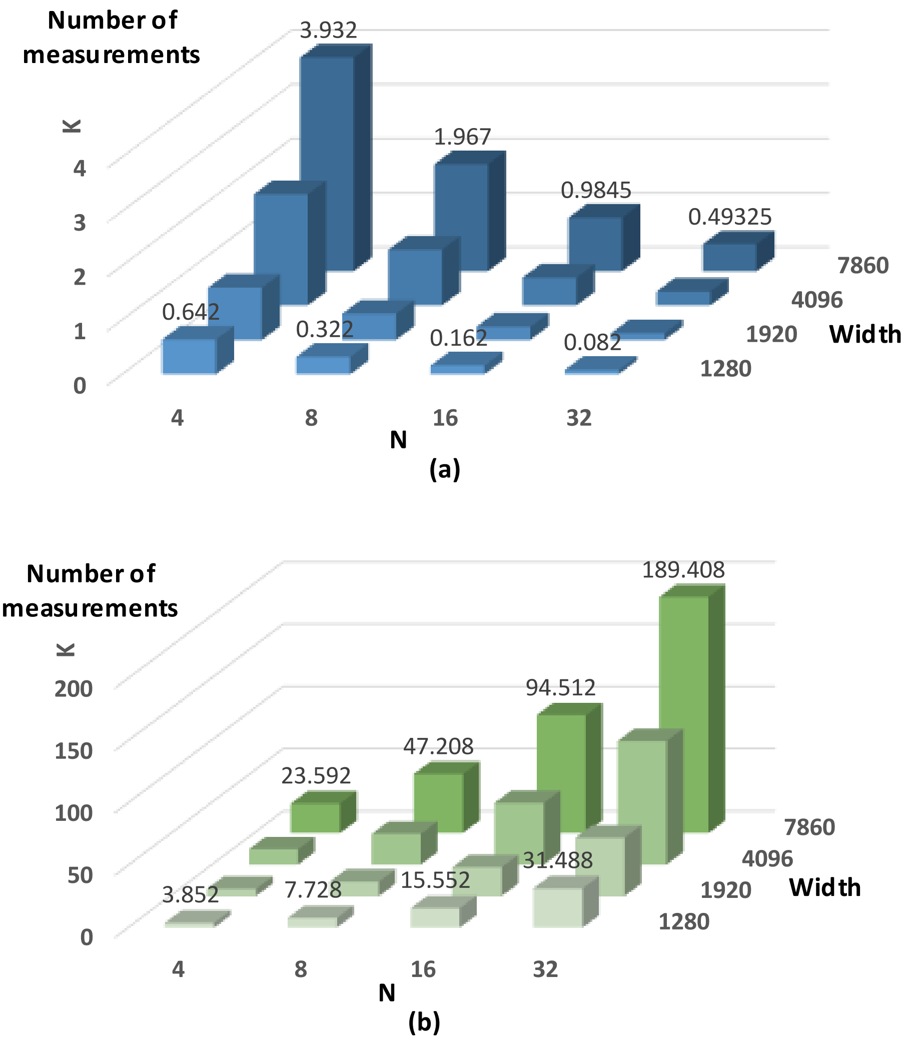


Fig. 37 Comparison of the number of measurements stored in the memory for intra prediction.
When SR ranging from 0.25 to 0.75. (a) This work. (b) SDPC [57].

4.5 Experimental results

The comparison is made among three algorithms, constant prediction (CP) as in [54] [55], SDPC [57] and our proposed algorithm. It is noted that the random binary [0/1] matrix is used as the measurement matrix in the first two methods. Based on this matrix, we modified the first two rows as we proposed in Section 4.3. Since the LSMM [59] does not apply the random [0/1] matrix, it is excluded in the comparison. Fourteen grayscale test images of size 512×512 are evaluated. The reconstruction algorithm is L1 primal– dual (PD) interior–point [51] with DCT.

First, the mean square error (MSE) is evaluated. The results with N = 4 and N = 16 are shown in the left three columns and right three columns, respectively, in Fig. 38. The MSE of each processing block in the proposed method is smaller and closer to zero compared with SDPC and CP. Moreover, the reduction in MSE between Prop. and CP is more significant for N = 4 than that for N = 16. Second, the bit rate is evaluated by entropy and the coding efficiency is evaluated by BD-PSNR [62], with the same method as in [57]. The BD-PSNR curves of four images are plotted in Fig. 39.

The results of coding efficiency of all images are shown in Table 13. Compared with CP, our proposed algorithm increases the coding efficiency by 2.33, 1.35, and 1.56 dB in BD-PSNR when N = 4, 8, and 16, respectively, equivalent to 46.6%, 27%, and 31.2% BD-rate reduction. Compared with SDPC, it also achieves increases in coding efficiency by 0.43, 0.44, and 0.29 dB in BD-PSNR when N = 4, 8, and 16 respectively, equivalent to 8.6%, 8.8%, and 5.8% BD-rate reduction. In accordance with the reduction in MSE, the reduction in bit rate shows that the small block has a higher compression ratio than the large block.

Since the measurement matrix is randomly generated, the occurrence of 1's in the measurement matrix has a great influence on the reconstruction quality. I find the optimal image quality can be achieved when occurrences of 1's are 74%, 23%, 4.305%, and 0.74% when N= 4, 8, 16, and 32, respectively. Moreover, we have tried other reconstruction algorithms, such as Total Variation minimization [51], which can also recover the image with similar image quality compared to the L1-PD algorithm. Furthermore, the visual quality comparison is shown in Fig. 40. There is no image quality degradation from the proposed algorithm. Meanwhile, it achieves bit rate reductions of 3% and 32% compared with SDPC and CP, respectively.

The performance of the proposed hardware is shown in Table 16. The total area is

9.1K, which includes 5.1K for SAD modules, 3.7K for intra prediction, and 0.3K for the finite state machine. For frame sizes of 4320 × 2160, the size of memory is 12 KB , growing linearly with the frame size. Since this is the first VLSI architecture for the measurement- domain intra prediction, there is no previous work to compare with. I roughly estimate that the SDPC would require 10.5K in area with 73.8KB SRAM, because the SDPC would require twice as many logic gates for SAD modules, six times as much memory to store the measurements and as much memory bandwidth to support the same throughput.

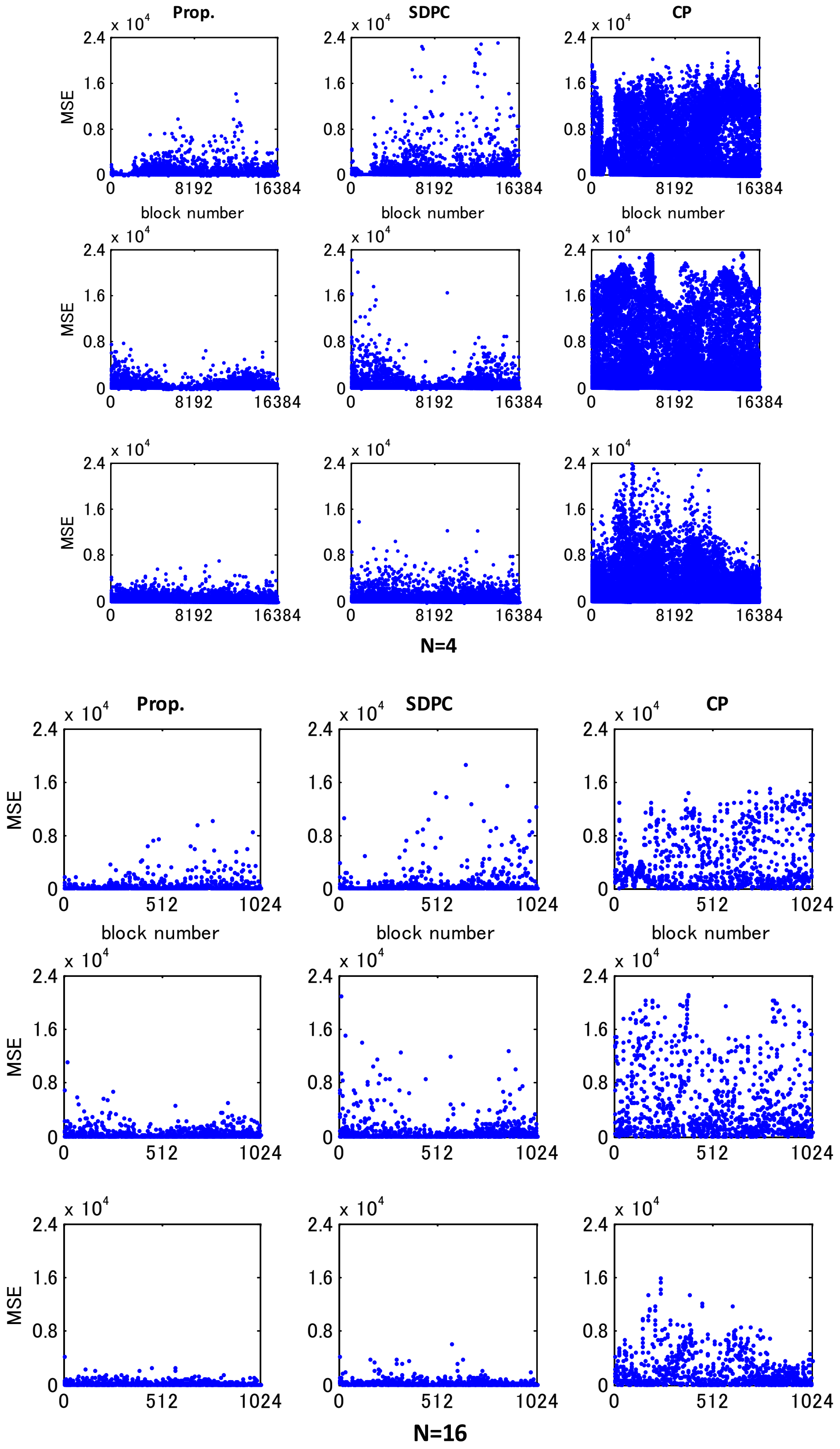


Fig. 38 Comparison of MSE of residuals in each block among three algorithms. With N = 4, 16 SR = 0.5, and Q_{step} =4 of lena (top row), barbara (middle row), and mandrill (bottom row)

Table 13 (a) BD-PSNR (BD-Rate) comparison. (Anchor is CP, with $Q_{step} \in [0,6]$ and N=4. Reconstruction algorithm is L1-PD with

DCT)

BD-PSNR (BD-Rate)						
N=4						
	SR=0.75		SR=0.50		SR=0.25	
Test Images	**Prop**	**SDPC**	**Prop**	**SDPC**	**Prop**	**SDPC**
Lena	6.116	5.581	2.699	2.043	0.469	0.825
Barbara	2.989	3.324	1.116	1.041	0.078	0.426
Mandrill	2.588	2.690	0.953	0.727	0.989	0.331
Peppers	2.452	1.684	1.445	0.321	0.032	0.122
house	6.937	6.015	4.779	1.547	0.086	0.443
F16	5.210	4.762	2.818	1.742	0.935	0.722
goldhill	5.516	5.197	2.485	1.739	0.752	0.841
pentagon	2.663	2.231	1.309	0.803	0.611	0.412
boat	4.163	3.786	2.035	1.243	0.467	0.534
bike	1.512	1.509	0.855	0.329	0.303	0.143
sailboat	3.772	3.447	1.907	1.011	0.769	0.444
milkdrop	8.761	8.173	4.081	2.493	1.133	1.588
elaine	6.185	5.656	3.160	2.232	1.330	1.051
Aver.	**4.528**	**4.158**	**2.280**	**1.329**	**0.612**	**0.606**

	Prop	**SDPC**
Aver. in all sample rate	**2.4733 (-33%)**	**2.0310 (-26%)**

Table 14 (b) BD-PSNR (BD-Rate) comparison.
(Anchor is CP, with $Q_{step} \in [0,6]$ and N=8. Reconstruction algorithm is L1-PD with DCT)

BD-PSNR (BD-Rate)						
N=8						
	SR=0.75		SR=0.50		SR=0.25	
Test Images	**Prop**	**SDPC**	**Prop**	**SDPC**	**Prop**	**SDPC**
Lena	2.806	2.384	1.181	0.555	0.941	0.161
Barbara	1.916	1.679	0.799	0.378	0.511	0.107
Mandrill	1.098	1.372	0.709	0.336	0.328	0.093
Peppers	1.460	0.907	0.590	0.190	0.888	0.056
house	6.713	6.420	2.403	1.589	1.445	0.432
F16	2.807	2.339	1.107	0.532	0.730	0.162
goldhill	2.731	2.196	1.105	0.611	0.758	0.207
pentagon	1.163	0.908	0.617	0.290	0.403	0.107
boat	1.976	1.630	0.906	0.408	0.756	0.136
bike	0.882	0.529	0.437	0.125	0.344	0.027
sailboat	1.683	1.385	0.824	0.320	0.652	0.100
milkdrop	4.914	4.711	1.315	1.166	1.056	0.310
elaine	2.645	2.113	1.154	0.619	0.756	0.232
Aver.	**2.523**	**2.198**	**1.011**	**0.548**	**0.736**	**0.164**

	Prop	**SDPC**
Aver. in all sample rate	**2.4733 (-28%)**	**2.0310 (-19%)**

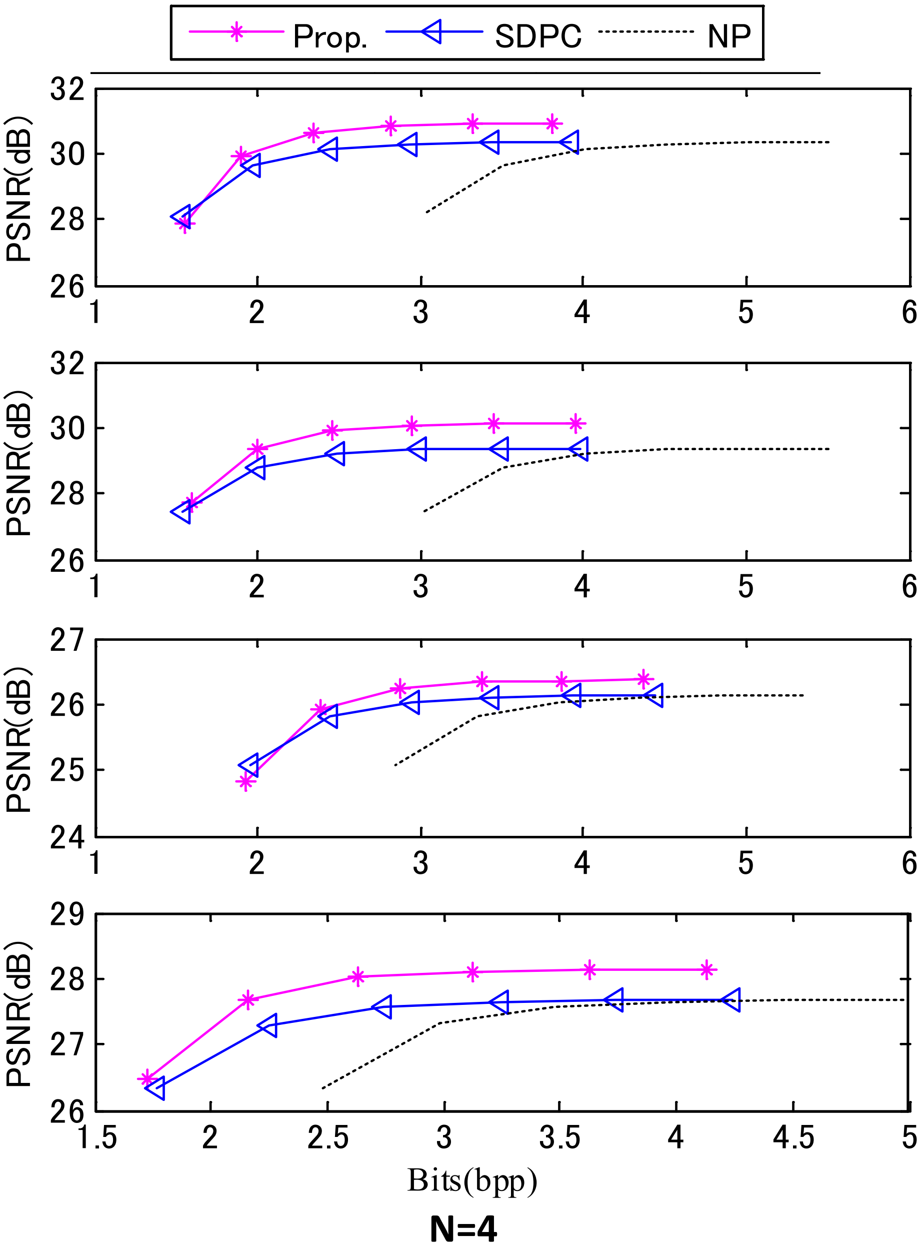
Table 15 (c) BD-PSNR (BD-Rate) comparison.
(Anchor is CP, with $Q_{step} \in [0,6]$ and N= 16. Reconstruction algorithm is L1-PD with DCT)

BD-PSNR (BD-Rate)						
N=16						
	SR=0.75		SR=0.50		SR=0.25	
Test Images	**Prop**	**SDPC**	**Prop**	**SDPC**	**Prop**	**SDPC**
Lena	3.526	2.952	1.143	0.796	0.750	0.221
Barbara	2.493	2.001	0.838	0.555	0.542	0.151
Mandrill	2.295	2.000	0.597	0.464	0.243	0.121
Peppers	2.843	2.432	0.963	0.630	0.696	0.173
house	8.354	8.136	2.902	2.458	1.279	0.656
F16	3.845	3.410	1.130	0.812	0.533	0.213
goldhill	3.110	2.873	1.009	0.854	0.550	0.270
pentagon	1.084	0.781	0.436	0.257	0.305	0.092
boat	2.303	2.026	0.776	0.574	0.490	0.174
bike	0.864	0.757	0.249	0.171	0.150	0.053
sailboat	2.203	1.953	0.667	0.477	0.468	0.117
milkdrop	6.279	6.286	1.931	1.810	0.876	0.440
elaine	2.872	2.478	1.067	0.774	0.788	0.259
Aver.	**3.236**	**2.930**	**1.054**	**0.818**	**0.590**	**0.226**

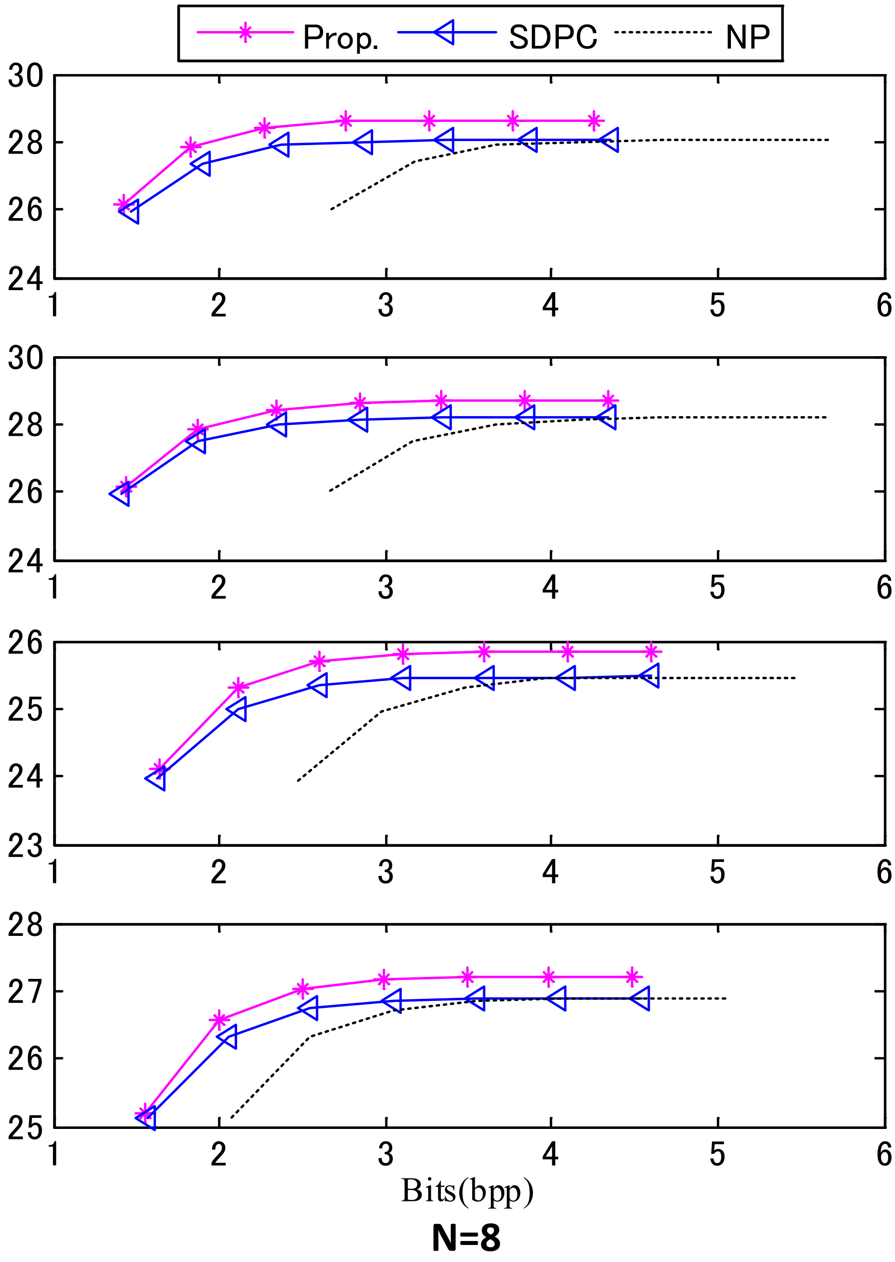
	Prop	**SDPC**
Aver. in all sample rate	**1.6269 (-49%)**	**1.3245 (-41%)**

Table 16 Performance of the architecture.

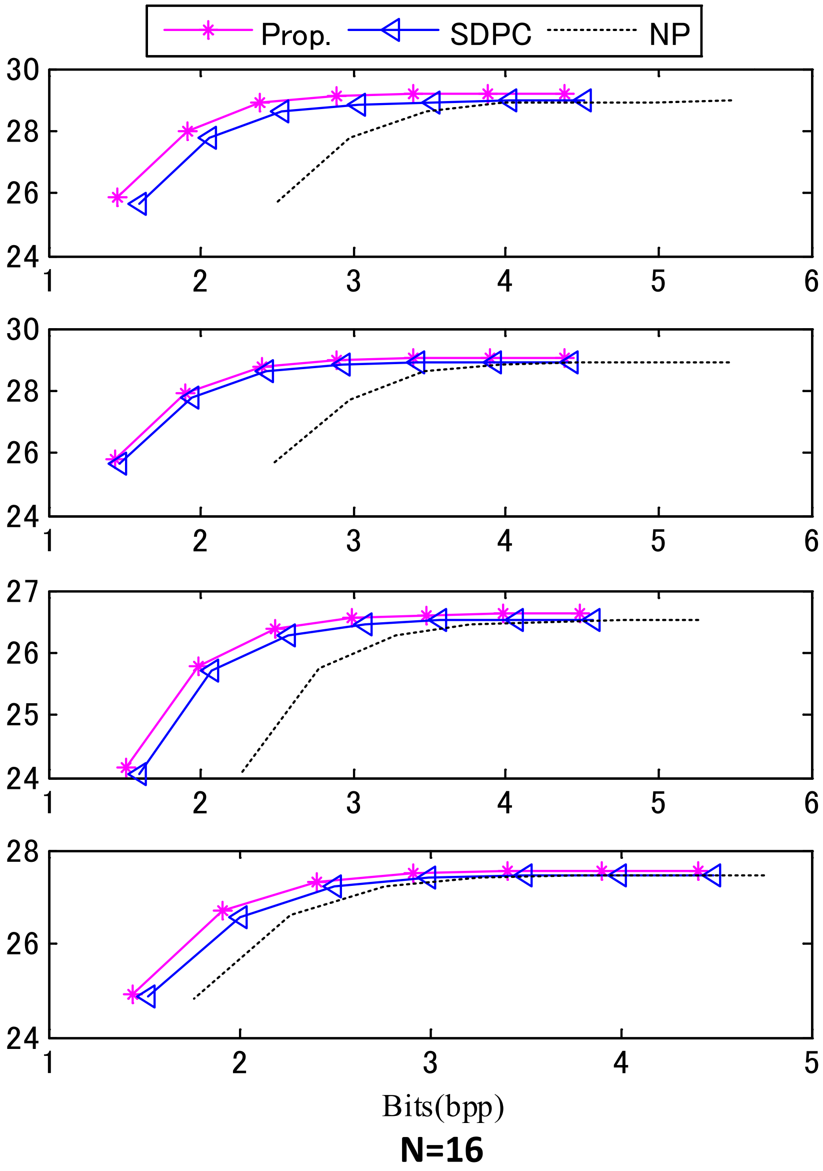
Process	SMIC 40nm
Area (Gates)	9.1K
Specification	4320p@240fps
Freq. (MHz)	200
SRAM	12 KB
Throughput (samples/Cycle)	16


Prop.
SDPC
NP
PSNR(dB)
32
30
28
26
1
2
3
4
5
6
27
25
24
29
1.5
2.5
3.5
4.5
Bits(bpp)
N=4

(a)



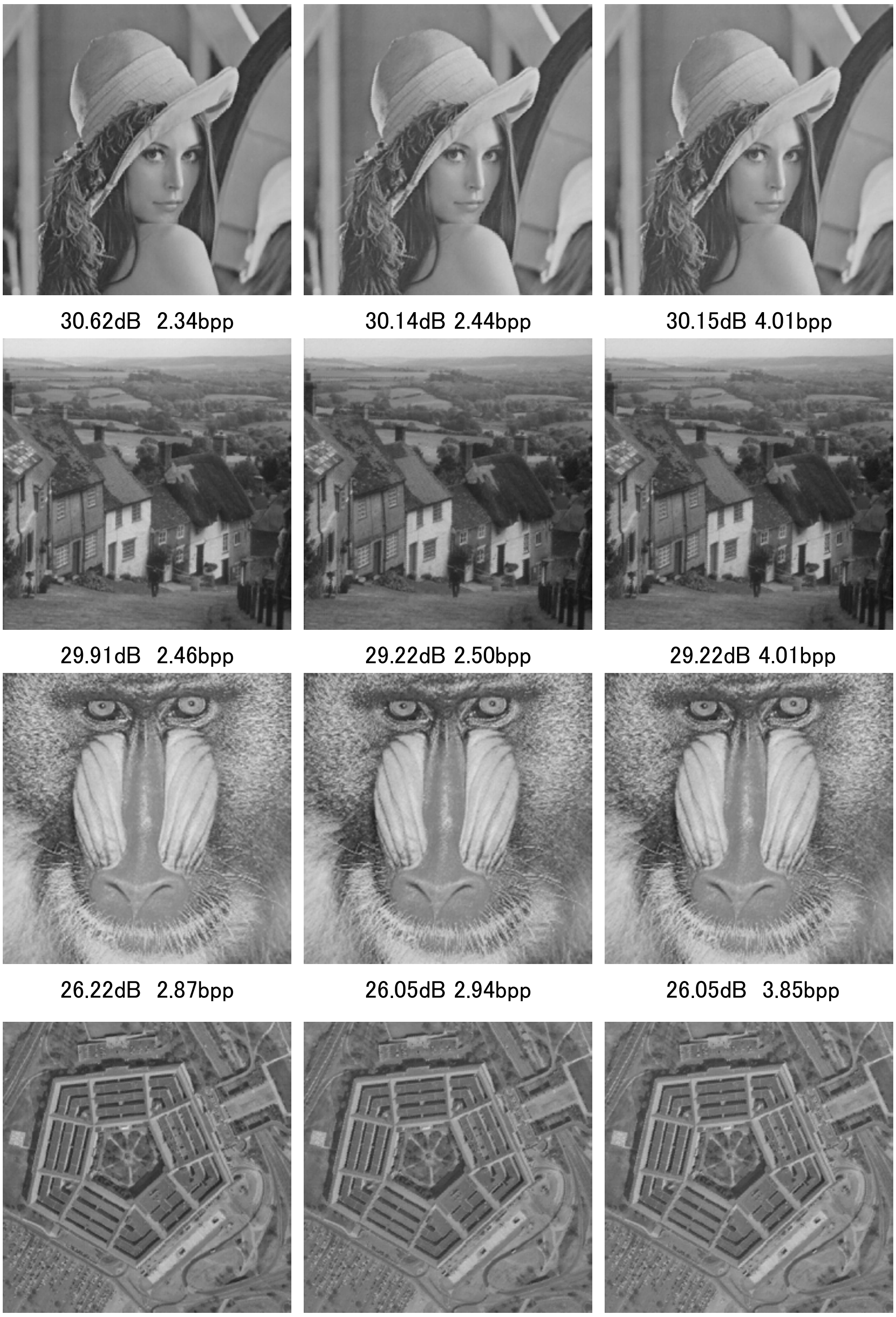
(b)



(c)

Fig. 39 BD-rate curve of three test images.

Lena (first row), goldhill (second row), mandrill (third row), and pentagon (fourth row) with N = 4 (a), 8 (b), and 16 (c), SR = 0.5, and Q_{step} varying from 0 to 6.



30.62dB 2.34bpp 30.14dB 2.44bpp 30.15dB 4.01bpp

29.91dB 2.46bpp 29.22dB 2.50bpp 29.22dB 4.01bpp

26.22dB 2.87bpp 26.05dB 2.94bpp 26.05dB 3.85bpp

28.03dB 2.64bpp 27.59dB 2.74bpp 27.59dB 3.48bpp

Fig. 40 Visual comparison among Prop., SDPC, and CP (left, middle, right). Four test images: lena, goldhill, mandrill, and pentagon with N = 4 and Q_{step} =4.

4.6 Summary

I proposed a measurements-domain intra prediction framework that is compatible with CS-based CMOS image sensors and shows low computational complexity. By artificially structuring two rows of the measurement matrix, the boundary information of neighboring blocks is embedded for intra prediction. Next, a low-cost VLSI architecture of the proposed framework was further proposed and implemented, by substituting the matrix multiplication with shared adders and shifter. The experimental results demonstrated that the VLSI architecture is 9.1K gates in area, and 12 KB dual-port SRAM memory. Working at 200 MHz, the architecture could support 4320p@240fps real-time encoding. The proposed framework could compress the measurements and increase coding efficiency, by 34.9% BD-rate reduction, and save up to 83% of the memory bandwidth and storage for line buffer and left neighboring buffer. It could significantly reduce both the energy consumption and the bandwidth in communication. The demerit of the proposed method is that two proposed predictors not always have good performance in all textures. The more precise intra predictor would be a future work.

5. Row-Operation-Based Intra prediction under Approximate-DCT measurement matrices and its VLSI Architecture implementation

5.1 Introduction

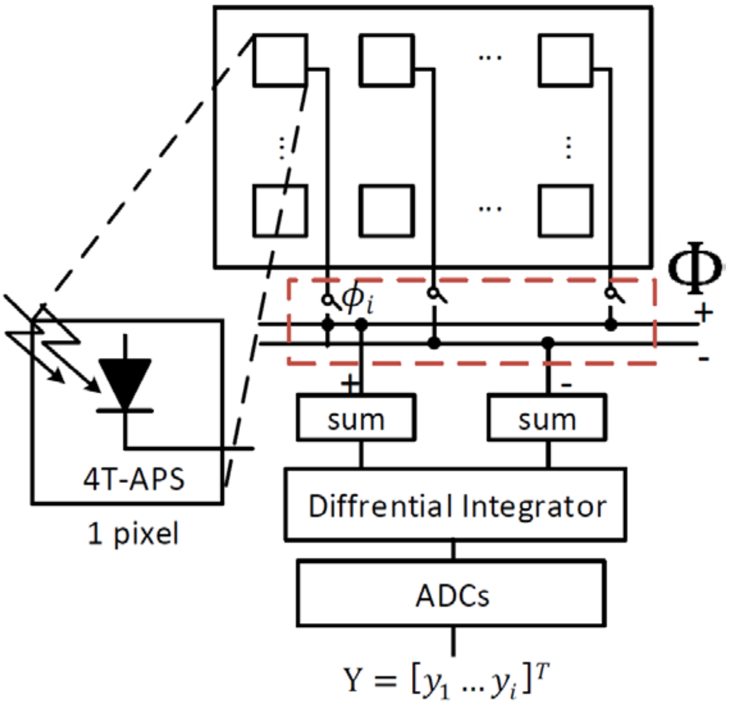


Fig. 41 A brief architecture of a processing block in CS-CIS [71] (Component A in Fig. 42). Outputs are digital measurements Y.

CMOS image sensor (CIS) has attracted a huge number of researches for the last decades. As most of the CIS applied in the mobile systems, the power consumption becomes a main concern. CIS first converts the analog luminance signal acquired into a digital one pixel by pixel, then compresses the image to reduce the data amount for the storage or for the further transmission, which is a capture (pixel)→compress (pixel) process. With the increase of resolution and frame rate in the recent years, however, the low-power design becomes a challenge. Since it is found that the Analog-to-Digital (A/D) conversions followed by the output readout is the main power consumption in

CMOS image sensor, increases linearly as least in resolution and frame rate. [54] With the advent of a recently proposed sampling theory, Compressed Sensing (CS) [51], an image could be acquired by capturing a significantly reduced number of measurements (the linear combination of pixels), instead of by capturing pixel by pixel. Such image sensors are called CS-based CIS (CS-CIS). The luminance signals are linearly combined into a measurement in analog domain, followed by the A/D conversion. Thus, the throughput of A/D conversion could be reduced, which results into a significant reduction in power consumption, as shown in the recent CS-CIS [54], [55], [71]. The output of CS-CIS – measurements are further compressed before the transmission. This is a capture (measurements) compress (measurements) process. In this measurement-based process, there're two issues to concern: how to increase the image quality and how to reduce the size of measurements. For the image quality, the measurement matrix plays a major role. It decides how pixels get combined into measurements. The binary/ternary measurement matrix is frequently used due to its simplicity in controlling the linear combination, which is achieved by the sum of current in in analog domain, as shown in Fig. 41. The binary/ternary matrix controls the switches to tell whether a pixel to be added or subtracted so that measurements are calculated by analog addition and differential integration [54], [55], [71]. However, the image quality of the binary/ternary measurement matrix being used is not satisfied, comparing with the Gaussian matrix. But the Gaussian matrix is not suitable for real implementation, because floating point elements in the matrix makes linear combinations hard to implement, requiring complex and energy-consuming analog multiplier. Moreover, several binary/ ternary measurement matrices proposed in [65] could outperform the Gaussian matrix a little bit, however, they can only be applied to sparse signals instead of directly to natural images, which means extra transform in CIS is required. Thus, a binary/ternary matrix could achieve high image quality is wanted.

To reduce the size of measurements, several previous works exploited the spatial redundancy to compress measurements. In [56], measurements in previous blocks were directly subtracted and used for prediction. Nevertheless, it only partly utilizes the horizontal correlation. In [57][58], the intra prediction occurs by the measurement-wise subtraction from the neighboring measurements, similar to pixel-wise subtraction inter prediction. In these works, however, the measurements for prediction contain irrelevant information, such as the

nonadjacent pixels, so that the prediction precision decreases. In [59], a local structural measurement matrix providing more precise prediction is proposed for the measurement-domain prediction by extracting the local features within a block. However, it has high computational complexity for a brute-force search among all the local predictor candidates, and it requires a floating-point measurement matrix that cannot be applied to the image sensor. Another issue in [57][58][59] is that they require all the measurements of a block for prediction. It requires a large memory bandwidth to fetch / load the data, as well as a large memory storage used for line buffer to store all the measurements of neighboring blocks, which would be a problem for a power-limited and a storage-limited wireless camera system. In [60][66], an intra prediction framework for measurement compression is proposed. It requires few memory storage and bandwidth, however, it needs to modify two rows of a random matrix, which might be not suitable to all matrices. Overall, these works could improve the coding efficiency, but the image quality still has spaces to improve.

In this paper, we therefore propose ternary measurement matrices to improve image quality and measurements compression algorithm as well as VLSI architecture to reduce the size of measurements. Main contributions of the paper are outlined as follows.

1) We proposed an algorithm to generate a series of deterministic and ternary measurement matrices, compatible to the current CS-CIS architecture [54], [55], [71]. The proposed matrices are derived from approximated DCT and capable to preserve the energy compact property as DCT. Comparing with random binary/ternary matrix, the proposed matrices achieve a significant improvement in image recovery quality and certain degree of bit rate saving.
2) We propose matrix row operations adaptive to the proposed matrix above for

measurements compression. It is able to generate the intra prediction pattern as our previous work [60][66], without constructing new rows, so that measurements could be further compressed without any image quality loss.

3) We implement hardware architecture of the proposed intra prediction for measurements compression presented above. The proposed matrices could simplified the architecture resulting into low hardware cost and low power consumption

5.2 Approximate DCT and Compressed Sensing (CS)

5.2.1 Approximated DCT:

DCT is a tool widely used in image compression due to its strong energy compaction property. However, it requires fast algorithm to reduce the computational complexity. Approximate DCT is one of the fast algorithm that offers a close result to exact DCT with hardware-friendly implementation.

The 2D-DCT of an image $\mathrm{R_o} = CR_iC^T$is approximated by $\widehat{\mathrm{R}}_\mathrm{o} = \hat{C}R_i\hat{C}^T$ where $\hat{C} = SP$ is an approximate matrix $(R_o, R_i, C, \hat{C}, S, P \in R^{N\times N})$. $S = \sqrt{(\mathrm{PP^T})^{-1}}$ is a diagonal matrix to orthogonalize $\hat{C}$. P is a coarse approximate DCT matrix with low-complexity that can even only consists of ternary numbers as 0/1/-1. Several works relate to the design of 4/8/16-point approximate DCT in [68],[69],[70]. A 4-point approximate matrix ($N = 4$) in [68] is shown as an example.

$$\mathrm{P} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & -1 \\ 1 & -1 & -1 & 1 \\ 0 & -1 & 1 & 0 \end{bmatrix} \tag{28}$$

5.2.2 Concept of CS:

The CS theory [51] asserts that only a few measurements are enough to recover the signals, as long as the signals are sparse in some transform domain. Suppose the image signal $\mathrm{X} = [\mathrm{x}_1 \ldots \mathrm{x}_n]^T$can also be represented in the transform domain Ψ, as

$$\mathrm{X} = \Psi\mathrm{S} \tag{29}$$

where $\mathrm{S} = [\mathrm{s}_1 \ldots \mathrm{s}_n]^T$ is the signal represented in Ψ transform domain and Ψ is an n × n transform matrix. The signal X is said to be k-sparse if it has only k non-zero coefficients.

I would like to recover signals $\mathrm{X} = [\mathrm{x}_1 \ldots \mathrm{x}_n]^T$ from $\mathrm{m} \ll \mathrm{n}$ linear and non-adaptive measurements $\mathrm{Y} = [y_1 \ldots y_m]^T$, which are taken from the random projection as

$$\mathrm{Y} = \Phi\mathrm{X} \tag{30}$$

where Φ is an m × n measurement matrix. I know that the system is under-determined since m < n. The CS theory asserts that the signal S' can be recovered with high probability using only $\mathrm{m} = \mathrm{cklog}(\frac{\mathrm{n}}{\mathrm{k}})$ measurements for some constant C, by solving the L1-norm minimization problem (3)

$$\min \|S'\|_1 \ \mathrm{s.t}\, \mathrm{Y} = \Theta \mathrm{S}' \tag{31}$$

where $\Theta = \Phi\Psi$ and the measurement matrix Φ must be in- coherent with transform matrix Ψ to preserve the Restricted isometry property (RIP) [51]. The CS theory shows that Φ can even be a random $1/-1$ or 0/1 matrix, while Ψ could be a discrete cosine transform (DCT), discrete wavelet transform (DWT), contourlet transform and so forth.

The problem (31) can be solved by basis pursuit [51]. To a noise environment, (31) can be extended to $\mathrm{Y}' = \Theta\mathrm{S}' + \mathrm{Z}$, where Z represents the noise. It could be solved by basis pursuit denoising [51]. After the recovery of S', the signal X' can thus be calculated by (29).

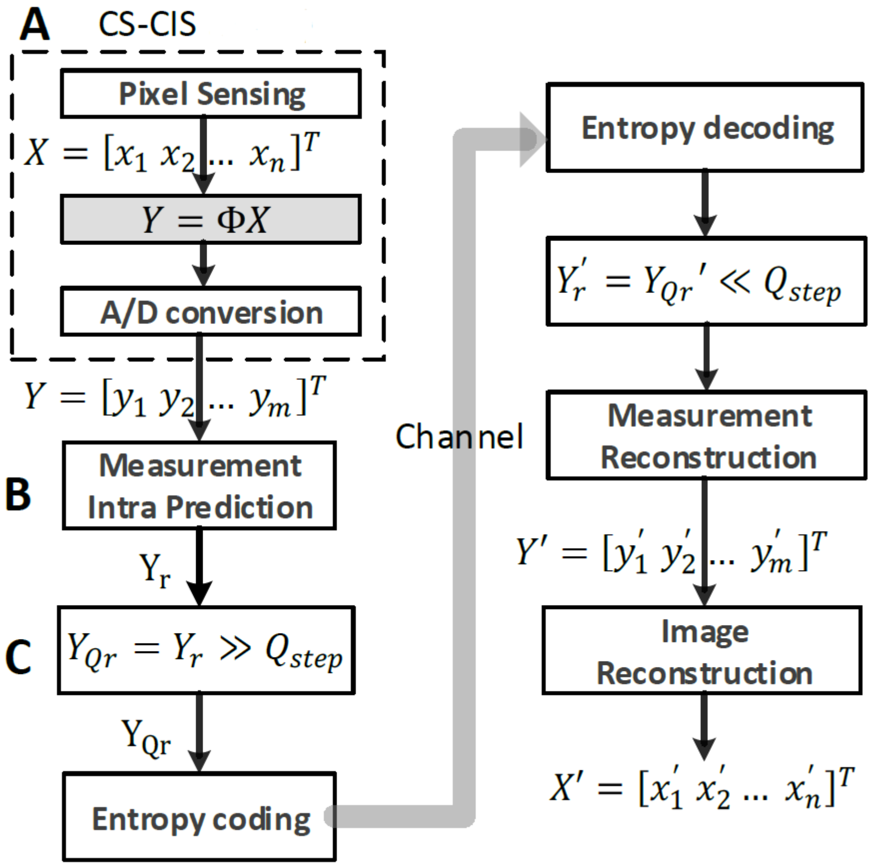

A
CS-CIS
Pixel Sensing
$X = [x_1\ x_2 \ldots\ x_n]^T$
$Y = \Phi X$
A/D conversion
$Y = [y_1\ y_2 \ldots\ y_m]^T$
B
Measurement Intra Prediction
$\mathrm{Y_r}$
C
$Y_{Qr} = Y_r \gg Q_{step}$
$\mathrm{Y_{Qr}}$
Entropy coding
Channel
Entropy decoding
$Y_r' = Y_{Qr}{}' \ll Q_{step}$
Measurement Reconstruction
$Y' = [y_1'\ y_2' \ldots\ y_m']^T$
Image Reconstruction
$X' = [x_1'\ x_2' \ldots\ x_n']^T$

Fig. 42 The process of CS based image sensing, encoding and decoding. The proposed measurement matrix is the gray box in component A. The proposedmeasurementsintrapredictionanditsarchitectureisincomponent B, C, shown in Fig. 8.

5.2.3 Process of CS image sensing and image reconstruction:

Considering the infeasibility and scalability of the image sensor implementation and the complexity of image recovery, the pixel array is divided into blocks as Fig. 43 (a) to perform the sampling [54][55]. For each block: 1) The analog signals of n pixels are acquired by the pixel array inside the sensor, $\mathrm{X} = [\mathrm{x_1} \ldots \mathrm{x_n}]^{\mathrm{T}}$. The measurements are calculated through (30) in the analog domain and digitalized into $\mathrm{Y} = [\mathrm{y_1} \ldots \mathrm{y_m}]^{\mathrm{T}}$. 2) Measurements are predicted, quantized, followed by the entropy coding and the transmission 3) The bitstream obtained from the channel are dequantized and then reconstructed into measurements. 4) The reconstructed signal X' are recovered from the decoded measurements Y' by solving (31) and (29). The process is shown in Fig. 42.

5.3 Proposed measurement matrix

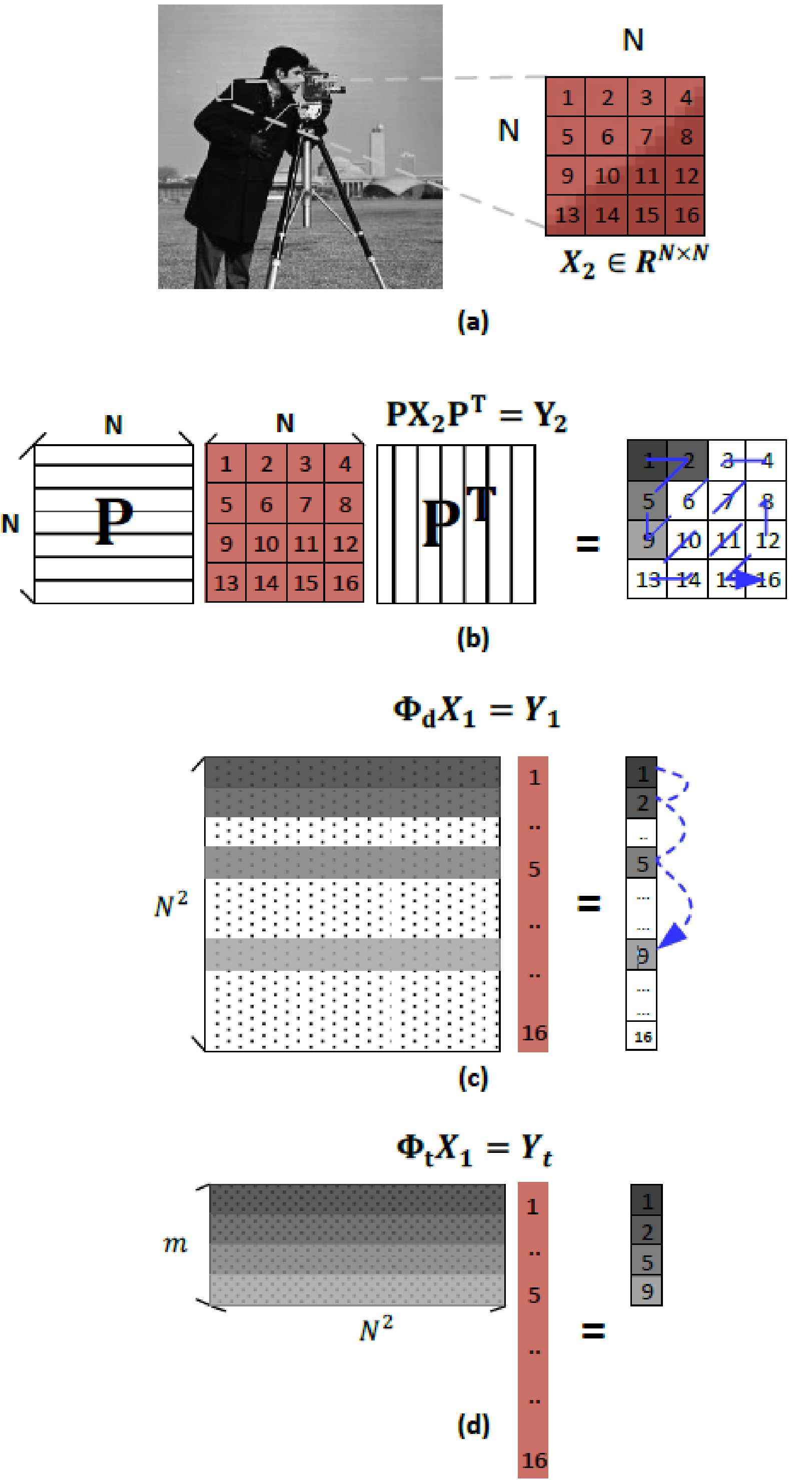


Fig. 43 (a) An image is separated block by block. (b) A 2D-DCT transform, N=4. The blue dashed line shows the Z-scan order, ascending frequency response. (c) The 1D representation of (b). (d) An example of N = 4, m = 4 measurements are taken in the Z-scan order..

5.3.1 The main idea

Because of the energy compact property of DCT, we propose a measurement matrix Φ_{t} that performs an approximated DCT to generate the measurements in (30). Measurements generated by the proposed matrix represent the frequency response of input signals, unlike the usual case that measurements are linear combination of input signals randomly taken. As is known, the low-frequency components in an image play the majority role in the determining the image quality. It gives us an intuition that measurements representing the lower frequency response would be more important than the ones representing the higher frequency response, if $m \ll N^2$ measurements are taken for image reconstruction.

Given any image signal $X \in R^{N^2}$, it could be represented by a 2D matrix $X_2 \in R^{N\times N}$ or an 1D matrix $X_1 \in R^{N^2\times 1}$. To perform the DCT on X, it can be a 2D-DCT as (32) in Fig. 43 (b), that sequentially performs 1D-DCT twice (vertically and horizontally), or can be a the projection to Φ_d, as (33) in Fig. 43 (c) ,such that Y_1, Y_2 represents the same output.

$$Y_2 = PX_2P^T \qquad (32)$$

$$Y_1 = \Phi_d X_1 \qquad (33)$$

where $P, Y_2 \in R^{N\times N}, Y_1 \in R^{N^2\times 1}$ and $\Phi_d \in R^{N^2\times N^2}$. The relationship between X_1, X_2 and Y_1, Y_2 can be represented by $X_1 = f(X_2)$ and $Y_1 = f(Y_2)$. The function f denotes a mapping from a 2D matrix M_2 to a 1D matrix M_1 , $f: {M_2}_{i,j} \rightarrow {M_1}_{(i-1)N+j}$.

In matrix Y_2 in Fig. 43 (b), the low frequency components are on the upper-left corner and the frequency increases according to the zigzag scan (Z-scan) order (blue dash line). Taking the lowest 4 frequency components (deep gray to light gray) as example, their corresponding locations in Y_1 are shown in Fig. 43 (c). Obviously, given any frequency component in Y_2, the corresponding measurements in Y_1 can always be located. Whenever to take $m \ll N^2$ measurements for image reconstruction, e.g $m = 4$ in this case, we can find the lowest m frequency components in Y_2 according the Z-scan order, and know which measurements to be kept in Y_1 . Since each measurement in Y_1 is determined by the corresponding row in Φ_d, as the gray area in Fig. 43 (c), we know which row to keep or given the number of measurement m. The final measurement matrix is shown in Fig. 43 (d).

Overall, there are two steps to generate the measurement matrix Φ_t : 1) Calculate the matrix Φ_d according to (34), where $i, j, k, q \in [1, N]$, p is an element of matrix P. 2) Trim the matrix Φ_d into the matrix Φ_t , by keeping the lowest m frequency components, which are the first m element in Z-scan order. It is noted that Φ_d is a ternary matrix, since the element p is ternary number.

$$\phi_{(i-1)N+j,(k-1)N+q} = p_{i,k}p_{j,q} \tag{34}$$

5.3.2 Derivation of proposed matrix Φ_d

Suppose there are matrices $D, X, E, G, Y \in R^{N\times N}$, such that $G_{i,j} = (DX)_{i,j}$ and $Y_{i,j} = (GE)_{i,j} = (DXE)_{i,j}$. According to the definition of matrix multiplication, $G_{i,j}$ and $Y_{i,j}$ can be expanded as (35) and (36)

$$G_{i,j} = \sum_{r=1}^{N} \mathrm{d_{i,r}} x_{r,j} = d_{i,1}x_{1,j} + d_{i,2}x_{2,j} + \cdots + d_{i,N}x_{N,j} \tag{35}$$

$$Y_{i,j} = \sum_{r=1}^{N} \mathrm{g_{i,r}} e_{r,j} = g_{i,1}e_{1,j} + g_{i,2}e_{2,j} + \cdots + g_{i,N}e_{N,j} \tag{36}$$

By observing some of the terms in (8) as follows, we expand (36)

$$g_{i,1} = \sum_{r=1}^{N} \mathrm{d_{i,r}} x_{r,1} = d_{i,1}x_{1,1} + d_{i,2}x_{2,1} + \cdots + d_{i,N}x_{N,N}$$

$$g_{i,2} = \sum_{r=1}^{N} \mathrm{d_{i,r}} x_{r,2} = d_{i,1}x_{1,2} + d_{i,2}x_{2,2} + \cdots + d_{i,N}x_{N,N}$$

$$\cdots$$

$$g_{i,N} = \sum_{r=1}^{N} \mathrm{d_{i,r}} x_{r,N} = d_{i,1}x_{1,N} + d_{i,2}x_{2,N} + \cdots + d_{i,N}x_{N,N}$$

$$Y_{i,j} = \sum_{r=1}^{N} \mathrm{g_{i,r}} e_{r,j} = g_{i,1}e_{1,j} + g_{i,2}e_{2,j} + \cdots + g_{i,N}e_{N,j}$$

$$= (d_{i,1}x_{1,1} + d_{i,2}x_{2,1} + \cdots + d_{i,N}x_{N,N}) + \big(d_{i,1}x_{1,1} + d_{i,2}x_{2,1} + \cdots + d_{i,N}x_{N,N}\big) + \cdots + (d_{i,1}x_{1,1} + d_{i,2}x_{2,1} + \cdots + d_{i,N}x_{N,N}) \tag{37}$$

By observing (37), we know each element in Y is a linear combination of $x_{1,1} \dots x_{N,N}$. Thus (37) could be simplified into

$$Y_{i,j} = \sum_{k,q=1}^{N} \left(d_{i,k} e_{q,j}\right) x_{k,q} \tag{38}$$

Let matrix $\dot{X}, \dot{Y} \in R^{N^2 \times 1}$ represent $X_{i,j}$ and $Y_{i,j}$where $\dot{X} = f(X)$ and $\dot{Y} = f(Y)$. Suppose

$$\dot{Y} = \Phi \dot{X} \tag{39}$$

The element $\dot{y}_{(i-1)N+j}$ is determined by the $[(\mathrm{i}-1)\mathrm{N}+\mathrm{j}]^{th}$ row of Φ and $\dot{X}$. For a given row in Φ, the element in each column determines the linear combination of $\dot{X}$, which is $d_{i,k} e_{q,j}$ in (38). Therefore, (38) could be represented as follows

$$\dot{Y}_{(i-1)N+j} = \sum_{k,q=1}^{N} \left(\phi_{(i-1)N+j,(k-1)N+q}\right) \dot{x}_{(k-1)N+q} \tag{40}$$

$$\phi_{(i-1)N+j,(k-1)N+q} = d_{i,k} e_{q,j} \tag{41}$$

where $\phi_{a,b}$ represents the element in row a and column b in Φ. By replacing D, E with $, P^T$, (41) becomes (34), and Φ d is the proposed matrix we want. By replacing $\dot{X}, \dot{Y}$ with X_1, Y_1, (39) becomes (33). We show Φ_d, Φ_t generated from (28) as an example. When $m = 4$, the $1^{st}, 2^{nd}, 5^{th}, 9^{th}$ rows (bold) are kept.

$$\Phi_d = \begin{bmatrix}
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\
\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{-1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{-1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{-1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{-1} \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} \\
1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\
1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 1 & -1 \\
0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\
\mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{-1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\
1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & -1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \\
0 & -1 & 1 & 0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & 1 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \tag{42}$$

$$\Phi_t = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1
\end{bmatrix} \tag{43}$$

5.3.3 The performance comparison

We compare the performance of two methods of trimming the measurement matrix, between Z-scan order (measurement index: 1,2,5,9 ...) as Fig. 43 (b) and normal-scan order (measurement index: 1,2,3,4 ...). From both graphs in Fig. 51, we can find that green curves occur in the lefter and upper region than the blue curves, showing that Z-scan order outperforms the N-scan order in different sizes of approximate matrix. The results verify our intuition that the preference for measurements of lowest m frequency response improves the image quality.

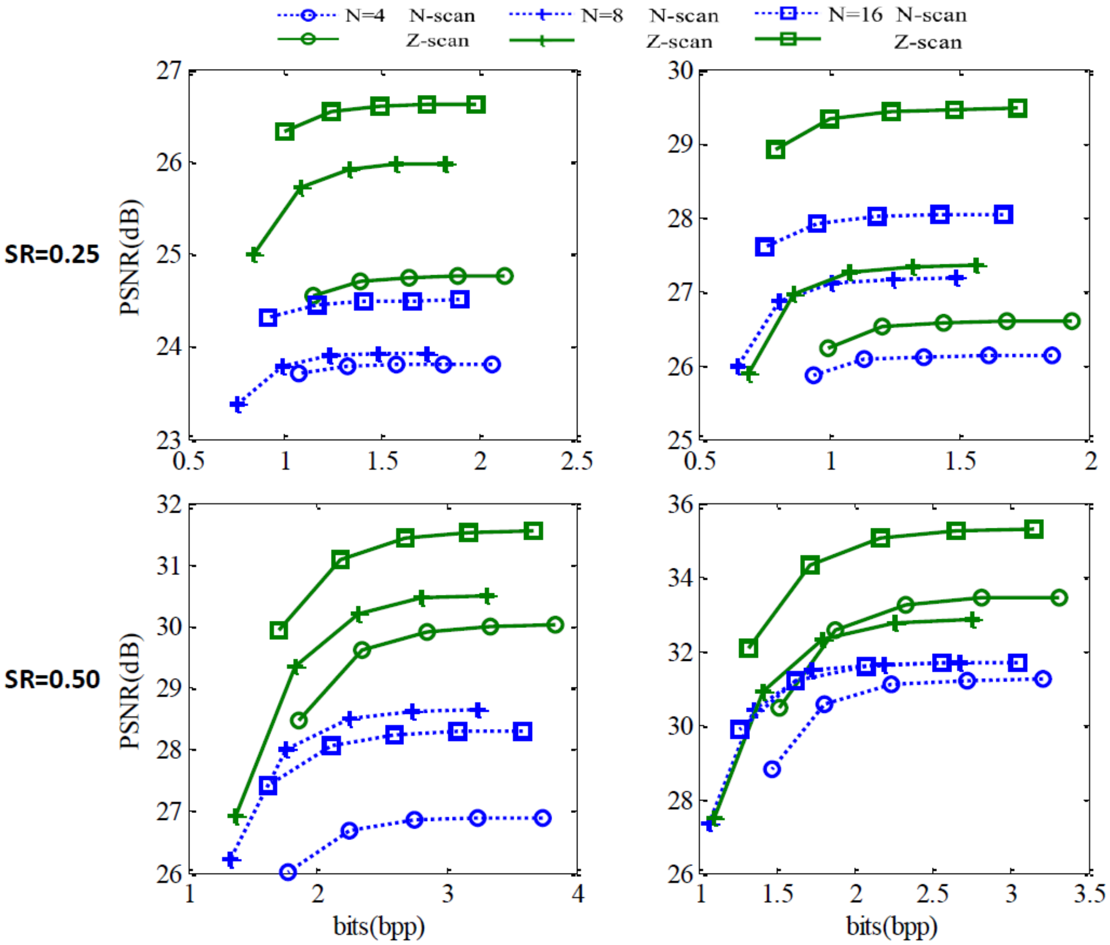


Fig. 44 The BD curve comparison between proposed measurement matrix with Z-scan and N-scan, using two images, mandrill (left) and F16 (right). Sample rate (SR) of 0.25 (first row) and 0.50 (second row) are evaluated. Matrices are generated the approximate DCT from several previous work (N=4: [68], 8: [69] and 16: [70]). The marker Circle, Plus and Square represent N=4,8 and 16 respectively. Blue dash line represents N-scan and green solid line Z-scan

5.4 Proposed matrix row operation for measurement-based intra prediction and its VLSI architecture

5.4.1 Existing measurement-based intra prediction

To reduce the data volume for storage and transmission, the measurements are further compressed. However, measurements, unlike pixels, could not be compressed by traditional intra coding methods, because the spatially correlation between adjacent pixels is corrupted during the generation of measurements. Thus, some works for measurements compression are proposed [57][58][60]. Among these works, [60] achieved the best coding efficiency of 7% BD-rate reduction.

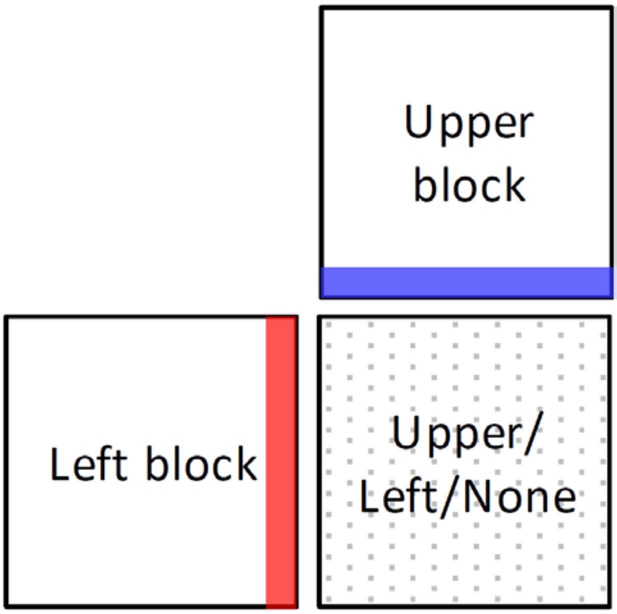


Fig. 45 The predictor candidates (Blue: Bottom row of upper block and red: rightmost column of left block)

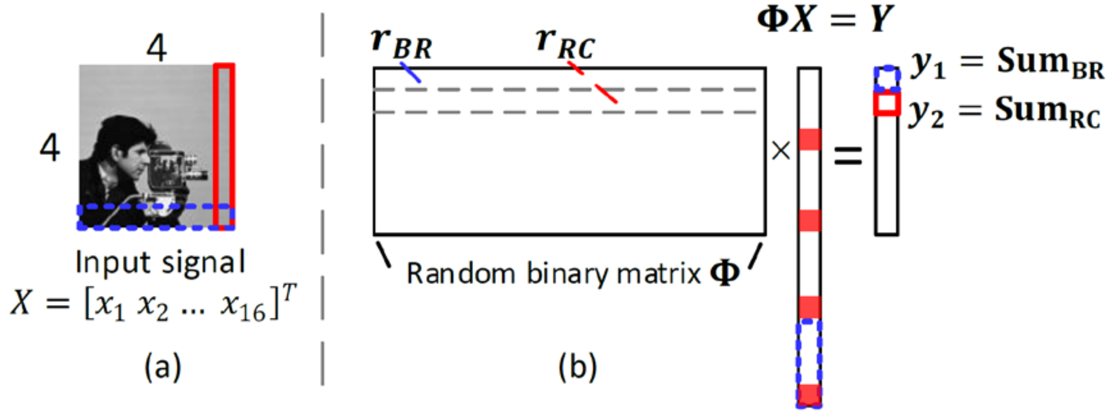

4
4
Input signal
$X = [x_1\ x_2\ \dots\ x_{16}]^T$
(a)
r_{BR}
r_{RC}
$\Phi X = Y$
$y_1 = \mathrm{Sum_{BR}}$
$y_2 = \mathrm{Sum_{RC}}$
×
=
Random binary matrix Φ
(b)

Fig. 46 A 4×4 block as input signal X. The target is to extract the sum of the blue and red part. (b) In the existing method [60][66], the first two rows in the random binary matrix are modified as r_{BR} and r_{RC} to extract the information (sum of bottom row and sum of rightmost column)

The basic idea of [60][66] is to extract the information of neighboring blocks (bottom row in upper block and rightmost column in left block) for predicting the current block, as shown in Fig. 45. Comparing with using pixels for prediction far away, using pixels nearby could improve the prediction accuracy. Thus, as shown in Fig. 46 (a), local information of a block (the sum of bottom row and sum of rightmost column) is extracted and stored when processing this block, so that next block could use them for prediction. To extract these local information, the first two rows of the random binary matrix are modified as r_{BR} and r_{RC} are shown in (44) and (45) when block size $N = 4$. In r_{BR}, the last N values are set to 1's, while the rest are set to 0's. In r_{RC} every N^{th} values are set to 1's, while the rest are set to 0's.

$$r_{BR} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1] \tag{44}$$

$$r_{RC} = [0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1] \tag{45}$$

When input signal X projects on these two modified row (the 1st and 2nd row), the resulted first two measurements y_1, y_2 would represent the sum of bottom row (Sum_{BR}) and sum of rightmost column (Sum_{RC}), which are the key information to be stored for future prediction.

5.4.2 Proposed matrix row operation for measurement-based intra prediction

However, the above approach [60][66] could not be applied to the proposed matrices in Section 3 without degrading the image quality, because it requires to modify two rows in the matrix to extract the information. As shown in (42) and (43), each row in the matrix is special, selected according to the frequency response. Changing a row could significantly degrades the image quality. Thus, our intuition is to find a way to extract the information of neighboring blocks as [60][66] without modifying proposed matrices. Two rows r_{BR} and r_{RC} are the keys to extract information of the bottom row and the rightmost column. According to our observation, we propose to use matrix row operation in the proposed matrices to calculate the target rows, as shown in Fig. 47.

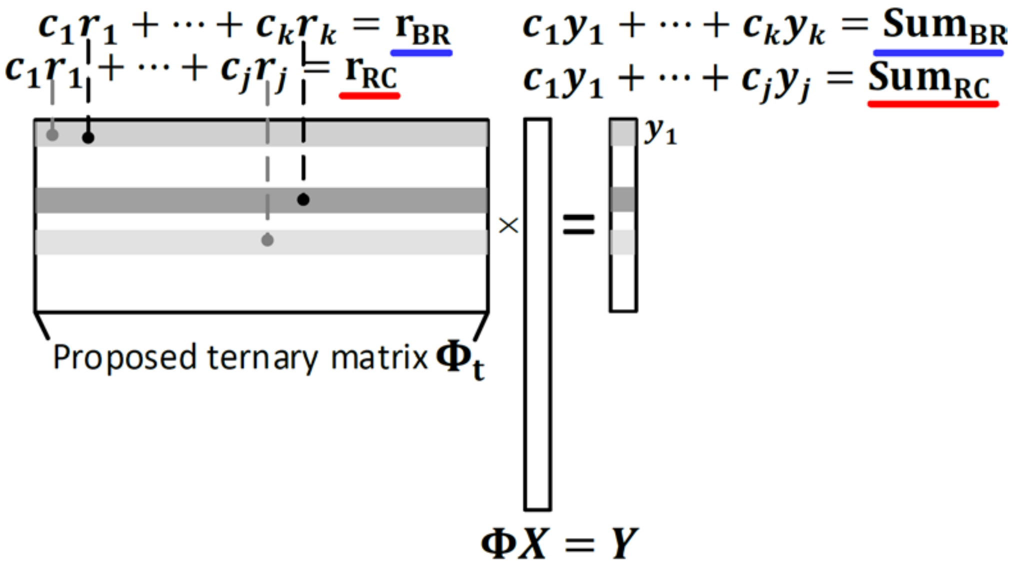

$c_1 r_1 + \cdots + c_k r_k = \mathbf{r}_{\mathbf{BR}}$
$c_1 r_1 + \cdots + c_j r_j = \mathbf{r}_{\mathbf{RC}}$
$c_1 y_1 + \cdots + c_k y_k = \mathbf{Sum}_{\mathbf{BR}}$
$c_1 y_1 + \cdots + c_j y_j = \mathbf{Sum}_{\mathbf{RC}}$
y_1
×
=
Proposed ternary matrix $\mathbf{\Phi}_{\mathbf{t}}$
$\mathbf{\Phi X = Y}$

Fig. 47 The proposed matrix row operation performed on the proposed matrix Φ_t to generate r_{BR} and r_{RC} to extract the sum of bottom row and the sum of rightmost column

Still using N=4 as an example, we generate a proposed matrix $\Phi_{t_6}$with m=6 in Z-scan order from Φ_d in (42), by taking the 1st, 2nd, 5th, 9th 6th and 3rd rows of Φ_d as the followings.

$$\Phi_{t_6} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 & 1 & 0 & 0 & -1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \end{bmatrix} \tag{46}$$

We observe that the key row r_{BR} in (44) could be obtained by matrix row operation from three rows, r_1, r_3, r_4 in the measurement matrix Φ_{t_6}, as (47).

$$r_1 = [1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1]$$

$$r_3 = [1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,-1-1-1-1]$$

$$r_3 = [1\,1\,1\,1-1-1-1-1\;-1-1-1-1\;-1-1-1-1]$$

$$(r_1 + \mathrm{r}_4 - 2 * \mathrm{r}_3) = 4 * \mathrm{r}_{\mathrm{BR}}$$

$$\mathrm{r}_{\mathrm{BR}} = (r_1 + \mathrm{r}_4 - 2 * \mathrm{r}_3) \gg 2 \tag{47}$$

Similarly, from matrix row operation from three rows, r_1, r_2, r_6 in the measurement matrix Φ_{t_6} , the key row r_{RC} in (45) could be obtained as (48)

$$r_2 = [1\,0\,0-1\,1\,0\,0-1\,1\,0\,0-1\,1\,0\,0-1]$$

$$r_6 = [1-1-1\,1\,1-1-1\,1\,1\;1-1-1\,1\,1\,1-1-1\,1\,1\,]$$

$$(r_1 + \mathrm{r}_6 - 2 * \mathrm{r}_2) = 4 * \mathrm{r}_{\mathrm{RC}}$$

$$\mathrm{r}_{\mathrm{RC}} = (r_1 + \mathrm{r}_6 - 2 * \mathrm{r}_2) \gg 2 \tag{48}$$

The discover above makes it possible to construct the key rows r_{BR} and r_{RC} without modifying the proposed measurement matrix Φ_{t_6}. Since each measurement is the projection of one row in the measurement matrix on the input signal, as mentioned in Section 4.1, the same matrix row operation performing on the measurements could obtain the neighboring information (the sum of bottom row and rightmost column) as the followings.

$$\mathrm{Sum}_{\mathrm{BR}} = (y_1 + \mathrm{y}_4 - 2 * \mathrm{y}_3) \gg 2 \tag{49}$$

$$\mathrm{Sum}_{\mathrm{RC}} = (y_1 + \mathrm{r}_6 - 2 * \mathrm{y}_2) \gg 2 \tag{50}$$

We have verified that the proposed matrix row operation could be applied to other approximate-DCT measurement matrices deriving from 8/16-point approximate-DCT matrix [69][70][70]. Since the proposed measurement matrix is determined, which rows to combine and how to combine to get the rows r_{BR}, r_{RC} could be obtained by offline calculation.

5.4.3 Measurement-based intra prediction framework and its VLSI implementation

After getting the Sum_{BR} and Sum_{RC} , the process of measurement-based prediction is the same as the one in [60][66], which could be divided into 4 steps. 1) Obtain the average of bottom row of upper block, $ave_{BR_{up}}$and obtain the average of rightmost column of left block, $ave_{RC_{le}}$, from Sum_{BR} and Sum_{RC}, respectively. 2) Use $ve_{BR_{up}}$, $ave_{RC_{le}}$and DC (eg. DC=128) to generate the predictor candidates Y_{up}, Y_{le}, Y_{DC} respectively, by performing the projection to the measurement matrix as (51), (52).

$$X_{up} = \left[ave_{BR_{up}}\ldots ave_{BR_{up}}\right]^T$$

$$Y_{up} = \Phi_t X_{up} \tag{51}$$

$$X_{le} = \left[ave_{RC_{le}}\ldots ave_{RC_{le}}\right]^T$$

$$Y_{le} = \Phi_t X_{le} \tag{52}$$

$$Y_{DC} = \Phi_t[DC \ldots DC]^T$$

3) Compare the sum and difference (SAD) of the original measurements Y and predictor candidates Y_{up}, Y_{le}, Y_{DC}, find the one with the minimum SAD as the predictor as (26) and use the original measurements to subtract it to get the residual as (54).

$$p = \mathrm{argmin}_{\mathrm{CAND}}\ \mathrm{SAD}\ (\mathrm{Y}, \mathrm{Y}_{CAND}) \tag{53}$$

$$Y_r = Y - Y_p \tag{54}$$

4) Quantize the residual by $Y_{Qr} = Y_r \gg Q_{step}$, followed by the entropy coding.

The VLSI architecture of the above measurement prediction is shown in Fig. 48. It is to compress measurements from CS-CIS (component A in Fig. 42) and then produces the quantized residuals Y_{Qr} to entropy coder. It takes measurements of the block size $N = 4$ [55] as input, with sample rate (SR) of 0.25, 0.5, and 0.75 (numbers of measurements m are 4, 8 and 12, respectively). A 4×4 block is processed every cycle. The major difference between [66] and the proposed architecture is that the matrix multiplication (the blue dash boxes) in this work has a significant lower hardware cost, because of the property of the proposed matrix Φ_d and Φ_t in (42) and (43). It could be observed that, the sum of element in each row of Φ_d and Φ_t is zero, expect for the first row (sum is 16, with sixteen 1's). Because the elements are identical in $X_{up}$and X_{le} respectively, when Φ_t multiplies the signal X_{up}, X_{le}, this property makes all measurements (except for the first one) equal to zeros. Thus, the matrix multiplication in (51) and (52) has not any computation, except for the first measurements (equal to 16*aves) requiring shift operation. Thus the matrix multiplication} nearly has no hardware cost, not even an adder. Besides, the property also reduces the calculations for the residual in (54), since only the first measurement needs subtraction, rather than all measurements. Though the proposed matrix row operation introduces the overhead in the red box at the bottom of Fig. 48, it has a low hardware cost, consisting of only 4 adders.

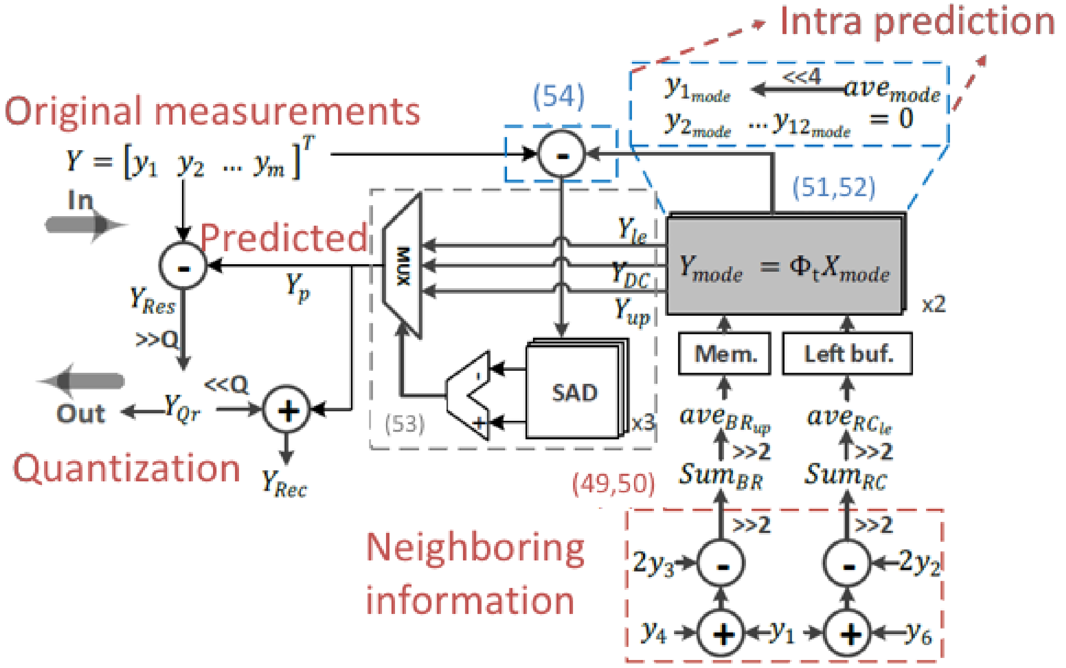


Fig. 48 VLSI architecture (Component B and C in Fig. 42) for the measurement-based intra prediction with proposed matrix row operation. The dashed boxes are marked by numbers in parentheses, corresponding to the equations (49)–(54) in Section 4. The matrix multiplication in the gray solid box could be simplified into the shift operation.

5.5 Experiment Results

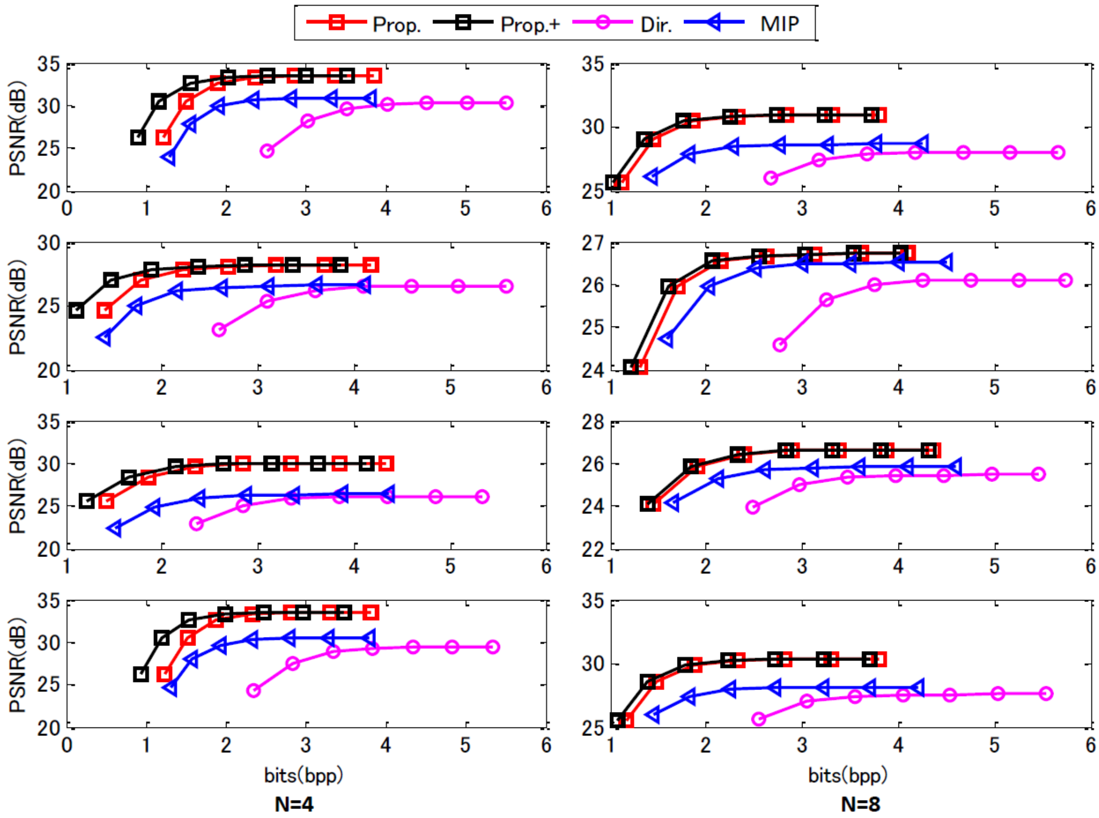


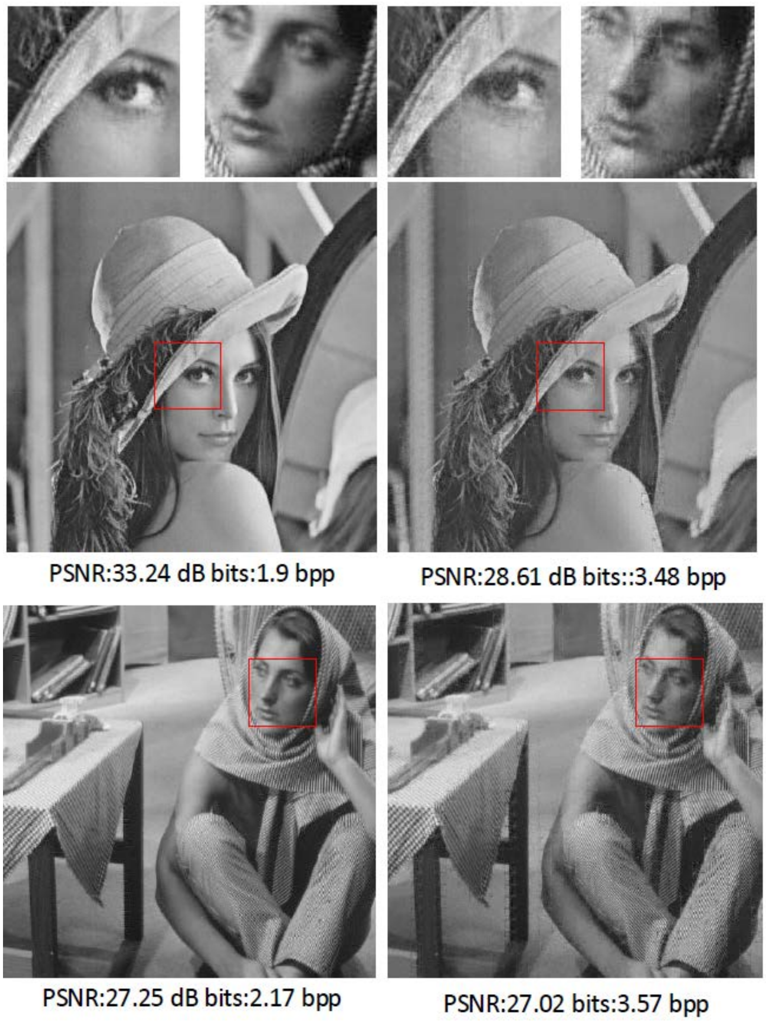
Fig. 49 The BD curve of lena, barbara, mandrill and F16 (top to bottom) with N=4 and N=8, SR =0.5, with various Q step ∈ [0, 6]. The proposed measurement matrix in Prop. and Prop.+ are trimmed by Z-scan.

The direct (Dir.) way [54],[55] and MIP [66], which use the random binary matrix (RBM), are compared with Prop. And Prop.+ in this work. In Dir., measurements are not compressed by any prediction method. In MIP, measurements are compressed by intra prediction. We define Prop. as the proposed matrix with Z-scan order in Section 5.3 but without any prediction. We define Prop.+ as the algorithm combining Prop. with the proposed matrix row operation for measurement intra prediction in Section 5.4.2. The performance of each method is evaluated under fourteen gray-scale test images (512 × 512), which are reconstructed by the algorithm, L1 primal-dual (PD) interior-point [51] with DCT.

First, the mean square error (MSE) is evaluated, as shown in Fig. 51. The results of block size $N = 4$ and $N = 8$ are in the left three columns and right three columns, respectively. The MSE of each processing block in Prop. are smaller and closer to zero than that in Dir. Prop.+ further decreases the MSE from Prop.. Moreover, the reduction in MSE between Prop. and Dir. more significant for $N = 4$ than that for $N = 8$. Next, we evaluate the BD-PSNR and BD-curve. Since the occurrence of 1's in the RBM used in MIP and Dir. influences the reconstruction quality. We find the optimal image quality can be achieved when occurrences of 1's are 74% when N=4 and 23% when N=8. From Fig. 49, we find that Prop. could significantly improve the image quality and reduce the size of measurements comparing with Dir., which is consistent with the result shown in Fig. 51. Prop.+ could reduce further reduce bit rate without introducing any image quality loss comparing with Prop. The result in Table 17 shows that Prop. could increase the BD-PSNR by 4.2 dB at average comparing with Dir., and 2.2 dB comparing with MIP. Prop.+ could further increase the BD-PSNR by 0.24 dB at average (equivalent to 5% BD-rate reduction) comparing with Prop. Finally, Fig. 50 shows an obvious improvement in image quality as well as in bit saving in Prop.+ on the left.

The performance of the hardware of Prop.+ (Component B in Fig. 42) and MIP are compared in Table 18. The total area of Prop.+ is 4.3 K gates, which includes 2.8 K gates for SAD modules, 1.2 K gates for intra prediction, 0.3 K gates for the finite state machine and the memory of 1 KB is for storing predictors. The area reduction is contributed by property of proposed matrix. It makes the matrix multiplication in intra prediction simpler. Its power consumption is 0.3 mW at 200MHz in typical condition (1.1 V, $25\,^{o}\mathrm{C}$). It is omittable (only 1%) comparing with the power consumption of the current CS-CIS [54][71], (28 mW to 100 mW). The throughput of our design is processing 12 measurements per cycle (a 4×4 block with SR = 0.75 has 12 measurements). The architecture could support 2160p@240fps.

Because of the proposed measurement matrix and measurement-based intra prediction, Prop.+ compresses measurements by 88% BD-Rate reduction comparing with Dir., with extra area of 4.3K gate and power consumption of 0.3 mW from the architecture. Comparing with MIP, Prop.+ also achieves compression of 49% BD-Rate reduction. Because we exploited the property of matrix to optimize the architecture, the area and power consumption of Prop.+ is 52% and 50% less than MIP, respectively.


PSNR:33.24 dB bits:1.9 bpp

PSNR:28.61 dB bits::3.48 bpp

PSNR:27.25 dB bits:2.17 bpp

PSNR:27.02 bits:3.57 bpp

Fig. 50 The visual quality comparison. Left :Prop.+ Right: Dir.

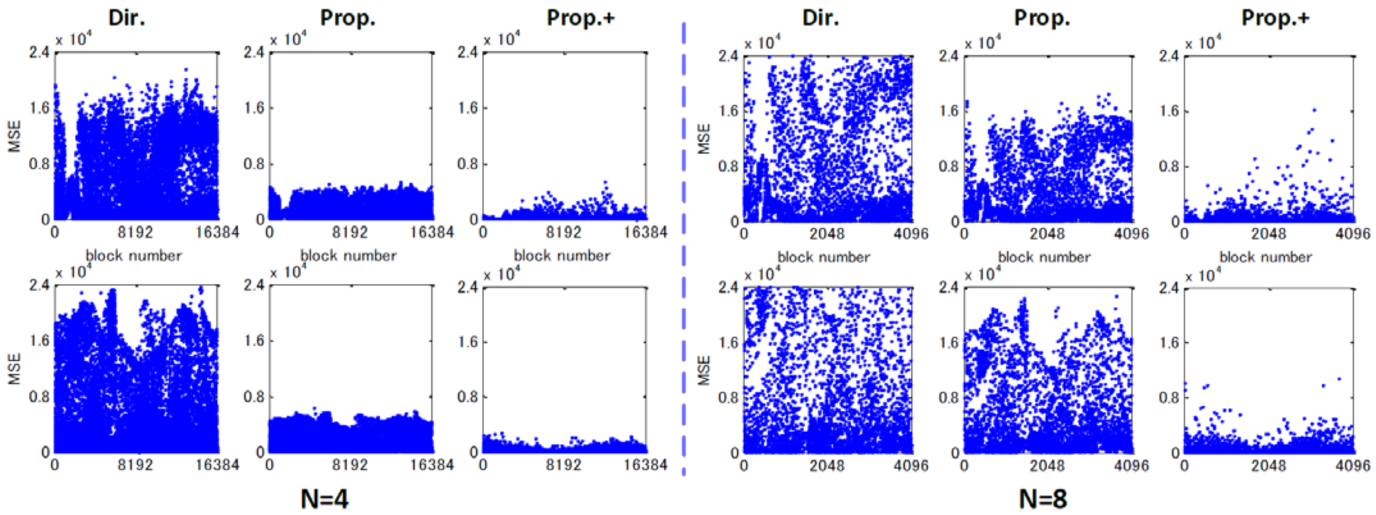


Fig. 51 Comparison of MSE of residuals in each block among three algorithms, with N = 4 and 8, SR = 0.5, and Q_{step} =4 of lena (first row) and mandrill (second row).

Table 17 BD-PSNR Comparison.

(Anchor is Dir., with $\mathbf{Q}_{step} \in [0,6]$, N=4 [13] and N=8 [69]. Reconstruction algorithm is L1-PD with DCT)

Test Images	**BD-PSNR**																	
	N=4									**N=8**								
	SR=0.75			**SR=0.50**			**SR=0.25**			**SR=0.75**			**SR=0.50**			**SR=0.25**		
	Prop.+	**Prop.**	**MIP**	**Prop.+**	**Prop.**	**MIP**	**Prop.+**	**Prop.**	**MIP**	**Prop.+**	**Prop.**	**MIP**	**Prop.+**	**Prop.**	**MIP**	**Prop.+**	**Prop.**	**MIP**
Lena	11.58	10.75	6.12	5.73	5.24	2.70	0.92	0.71	0.47	7.40	7.28	2.81	3.70	3.65	1.18	1.54	1.53	0.94
Barbara	4.86	4.40	2.99	2.85	2.65	1.12	0.26	0.17	0.08	3.27	3.19	1.92	1.11	1.09	0.80	0.02	0.01	0.51
Mandrill	7.08	6.79	2.59	4.75	4.64	0.95	2.06	2.02	0.99	4.02	3.98	1.10	1.56	1.55	0.71	1.24	1.23	0.33
Peppers	7.47	6.69	2.45	4.82	4.63	1.44	0.44	0.37	0.03	5.37	5.28	1.46	3.04	3.01	0.59	1.58	1.57	0.89
house	18.37	15.86	6.94	7.76	6.51	4.78	0.24	0.04	0.09	14.79	14.47	6.71	5.72	5.53	2.40	1.80	1.74	1.45
F16	11.56	10.85	5.21	6.16	5.79	2.82	1.48	1.36	0.93	7.61	7.49	2.81	3.43	3.39	1.11	1.69	1.68	0.73
goldhill	10.58	9.87	5.52	5.60	5.24	2.48	1.14	1.01	0.75	6.28	6.19	2.73	2.96	2.92	1.11	1.56	1.55	0.76
pentagon	6.38	6.15	2.66	3.47	3.37	1.31	1.24	1.19	0.61	3.46	3.44	1.16	1.91	1.89	0.62	1.24	1.22	0.40
boat	8.75	8.25	4.16	5.19	4.97	2.03	0.61	0.53	0.47	5.33	5.26	1.98	3.02	3.00	0.91	1.73	1.72	0.76
bike	4.68	4.34	1.51	2.98	2.88	0.85	0.76	0.73	0.30	2.57	2.53	0.88	1.39	1.38	0.44	0.92	0.92	0.34
sailboat	8.09	7.56	3.77	4.85	4.64	1.91	1.34	1.26	0.77	5.16	5.09	1.68	2.95	2.93	0.82	1.64	1.63	0.65
milkdrop	15.98	14.68	8.76	9.35	8.58	4.08	0.84	0.45	1.13	9.96	9.71	4.91	4.52	4.39	1.32	1.69	1.64	1.06
elaine	8.89	8.11	6.19	4.93	4.43	3.16	1.77	1.54	1.33	5.78	5.69	2.64	3.20	3.16	1.15	1.32	1.31	0.76
Aver.	**9.56**	**8.79**	**4.53**	**5.27**	**4.89**	**2.28**	**1.01**	**0.88**	**0.61**	**6.23**	**6.12**	**2.52**	**2.96**	**2.91**	**1.01**	**1.38**	**1.36**	**0.74**
Aver. in all sample rate	**Prop.+**			**Prop.**			**MIP**			**Prop.+**			**Prop.**			**MIP**		
	5.28			4.85			2.47			3.53			3.47			1.42		

Table 18 Performance of VLSI architecture.

	Prop.+	**MIP [66]**
Process	SMIC 40nm	
Area (NAND Gates)	4.3 K	9.1 K
Specification	2160p @ 240fps	
Freq. (MHz)	200	
SRAM	1 KB	
Throughput (measurements/Cyc.)	12	
Power Consumption	0.3 mW	0.6 mW

5.6 Summary

We proposed an algorithm to generate a series of deterministic and ternary matrices, which are compatible with the CS-CIS. The proposed matrices are derived from the approximate DCT, hence preserving the energy compaction property as DCT does. The proposed measurement matrix significantly improves the coding efficiency by BD-PSNR increase of 4.2 dB, comparing with the random binary matrix used in the-state-of-art CS-CIS. We further proposed matrix row operations adaptive to the proposed matrix to compress measurement by 4.8% BD-rate without any image quality loss. Lastly, a low-cost and low-power VLSI architecture of the proposed measurements intra prediction is implemented, with only 4.3 K gates in area, 0.3 mW in power consumption and supporting 2160p@240fps.

6. Conclusions and future work

This dissertation discusses the high-performance VLSI architecture of HEVC SAO Estimation, intra prediction for encoder and its extension in Compressed Sensing, by using the proposed concept “reduced video data”. Only by taking the necessary video data, including pixels and measurements, it is possible to reduce the parallel degree in hardware while keeping the performance during the data processing. The whole dissertation is organized into four parts, where the first two parts are intra prediction and SAO in HEVC, while the third and fourth are the HEVC intra prediction’s application to CS.

Firstly, in Chapter 2, the high-performance VLSI architecture for HEVC intra prediction is presented. Intra prediction uses neighboring pixels from different directions to predict pixels of a block (4x4~32x32). As the block size increases from 16 to 32 in HEVC, it takes 3x more neighboring pixels for prediction. Instead of loading all neighboring pixels as previous work, only on-demand pixels are loaded. This proposed idea reduces the two-third of reference pixels, thus reducing the area and increasing the throughput. It is achieved by LUT generated by software to tell which pixels are demanded in each prediction mode and location. Another proposal is the Hybrid Block Reordering and Data Forwarding, minimizing the idle time and eliminating the dependency between blocks by creating 3 Data Forwarding paths. It achieves the hardware utilization of 94%. The proposed VLSI architecture has a gate count of 217.8K, able to support 4320p@120fps HEVC intra prediction. The demerit of the proposal is that the bandwidth of SRAM is increased, as multiple loadings of reference samples are required for a block larger than 4x4. This would be a problem to be solved in the future work.

Next, in Chapter 3, the VLSI Architecture for SAO estimation is proposed. SAO estimation consists of two processes, statistics collection (SC) and parameter decision (PD), each of which demands different frequency. After investigating the optimal frequency, a dual-clock architecture is proposed to deal with SC and PD with different speed of clocks. Such a strategy reduces the overall area by 56%. To further improve the area and power efficiency, algorithm-architecture co-optimizations are applied including a coarse range selection (CRS) and an accumulator bit width reduction (ABR). CRS shrinks the range of fine processed bands for the band offset estimation. ABR further reduces the area by narrowing the accumulators of SC. They together achieve another 25% area reduction. The proposed VLSI design is capable of processing 8K@120fps encoding. It occupies 51K logic gates, only one-third of the circuit area of the state-of-the-art design. The demerit of the proposal is that the local heat problem would exist due to the high frequency clock.

Furthermore, in Chapter 4, a measurement intra prediction framework and its VLSI architecture are presented. Instead of using all measurements for prediction, measurements for prediction are reduced to two. These two measurements embed the block boundary information of closest area. They are obtained by modifying two rows in the random 0/1 measurement matrix. Furthermore, a low-cost VLSI architecture is implemented for the proposed framework, by substituting the matrix multiplication with shared adders and shifters. The experimental results show that our proposed framework can compress the measurements and increase coding efficiency, with 34.9% BD-rate reduction compared to the direct output of CS-based sensors. The VLSI architecture of the proposed framework is 9.1K in area, and it achieves the 83% reduction in size of memory bandwidth and storage for the line buffer. This could significantly reduce both the energy consumption and bandwidth in communication of wireless camera systems. The demerit of the proposed method is that two proposed predictors not always have good performance in all textures.

At last, in Chapter 5, a series of deterministic and ternary matrices derived from approximated-DCT are proposed, which could be used as measurements matrices. They significantly increase the coding efficiency comparing with the random binary matrix in previous work. Furthermore, an algorithm using the row-operation to perform the intra prediction on the approximate-DCT measurement matrices is proposed. Without modifying the measurement matrix to structure the specific row as previous work in

Chapter 4, similar effects could be achieved by the row-operation of specific measurements. Lastly, a low-cost VLSI architecture of measurements compression with proposed matrix row operations is proposed. Experiment results show the proposed matrix improve the coding efficiency by BD-PSNR increase of 4.2 dB. The proposed row operations increase the coding efficiency by 0.24 dB BD-PSNR. The VLSI architecture is only 4.3 K gates in area and 0.3 mW in power consumption. The demerit is that some of the row-operations are not available in low sampling rate, because it relies on the measurements which do not exist in the low sampling rate.

The future work includes the following aspects. First is how to design a more efficient reference samples loading and storing scheme, that could further reduce bandwidth of SRAM. Second is how to further explore the possibility to implant the HEVC intra prediction to CS to further improve its coding efficiency. At last, there’s lots of spaces for studying the extension of HEVC inter prediction to CS.

Acknowledgement

First and foremost, I would like to thank my main advisor, Professor Jiro Katto. I am thankful for his kindness to me and for accepting to be my main advisor. He gives me valuable idea of the writing of the thesis and valuable supports. I would like to thank my advisor of master period, Professor Satoshi Goto, who has given me lots of guidance in my study and life. He is such an attractive person who I admire as an idol and would like to become a person like him in my future. I would also like to thank my advisor of the PhD period in IPS, Professor Takeshi Yoshimura, who is always speechless but actually so kind to me. He gave me a great deal of supports for the past three years. Especially He taught me how to explain the idea more clearly and more concise in the presentation and in the thesis. I also thank Professor Nozomu Togawa for his precious advices for improving the presentation and the thesis.

I would also like to express my special gratitude to Dr. Dajiang Zhou, who acts as my master as well as my friend. He has been teaching me most of the academic skills I have, taught me how to be a researcher and has given lots of guidance in daily life.

I am also very grateful to my friend Cheng Chen, Shihao Wang, Yingjun Wu, Yijin Zhao, Li Guo, Jinjia Zhou, Kaiyi Yang, Shuping Zhang, Jiayi Zhu, Heming Sun, Fan Wang and Haoming Zhang for the days we spent together for research discussion, life sharing for the past several years.

I would like to thank my parents, my aunts, my god father passing away two years ago for their endless supports for the past five years.

Finally, I would like to thank the Program for the Leading Graduate Schools, “Graduate Program for Embodiment Informatics” of the Ministry of Education, Culture, Sports, Science and Technology.

References

[1] H. C. Andrews and W. K. Pratt, “Fourier transform coding of images,” in Proc. Hawaii Inr. Conf. Svst. Sci., pp 677-678, Jan. 1968.

[2] W. Chen and W. Pratt, "Scene Adaptive Coder," in IEEE Transactions on Communications, vol. 32, no. 3, pp. 225-232, Mar 1984.

[3] J. O. Limb, R. F. W. Pease and K. A. Walsh, "Combining intra frame and frame-to-frame coding for television," in The Bell System Technical Journal, vol. 53, no. 6, pp. 1137-1173, July-August. 1974.

[4] P. List, A. Joch, J. Lainema, G. Bjontegaard and M. Karczewicz, "Adaptive deblocking filter," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 7, pp. 614-619, July 2003.

[5] A. Norkin et al., "HEVC Deblocking Filter," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1746-1754, December 2012.

[6] “H.265: High efficiency video coding,” ITU-T Rec, 2013.

[7] G.J. Sullivan, J. Ohm, W.J. Han, and T. Wiegand, “Overview of the high efficiency video coding (HEVC) standard,” Circuits and Systems for Video Technology, IEEE Transactions on, vol.22, no.12, pp.1649–1668, 2012.

[8] “Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264 — ISO/IEC 14496- 10 AVC),” ITU-T Rec, 2003.

[9] T. Ito, “Future television super hi-vision and beyond,” Solid State Circuits Conference (A-SSCC), 2010 IEEE Asian, pp.1–4, IEEE, 2010.

[10] F. Li, G. Shi, and F. Wu, “An efficient VLSI architecture for 4x4 intra prediction in the High Efficiency Video Coding (HEVC) standard,” Image Processing (ICIP), 2011 18th

IEEE International Conference on, pp.373–376, September 2011.

[11] C.T. Huang, M. Tikekar, and A. Chandrakasan, “Memory- hierarchical and mode-adaptive HEVC intra prediction architecture for quad full HD video decoding,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol.22, no.7, pp.1515–1525, July 2014.

[12] H. Jung and K. Ryoo, “An intra prediction hardware architecture with low computational complexity for HEVC decoder,” in Future Information Communication Technology and Applications, pp.549– 557, Springer, 2013.

[13] D. Palomino, F. Sampaio, L. Agostini, S. Bampi, and A. Susin, “A memory aware and multiplierless VLSI architecture for the complete intra prediction of the HEVC emerging standard,” Image Processing (ICIP), 2012 19th IEEE International Conference on, pp.201–204, IEEE, 2012.

[14] C. Liu, W. Shen, T. Ma, Y. Fan, and X. Zeng, “A highly pipelined VLSI architecture for all modes and block sizes intra prediction in HEVC encoder,” ASIC (ASICON), 2013 IEEE 10th International Conference on, pp.1–4, IEEE, 2013.

[15] N. Zhou, D. Ding, and L. Yu, “On hardware architecture and pro- cessing order of HEVC intra prediction module,” Picture Coding Symposium (PCS), 2013, pp.101–104, IEEE, 2013.

[16] G. He, D. Zhou, J. Zhou, T. Zhang, and S. Goto, “A 530mpixels/s intra prediction architecture for ultra high definition h.264/avc encoder,” IEICE Transactions on Electronics, vol.E94.C, no.4, pp.419– 427, 2011.

[17] J. Zhou, D. Zhou, H. Sun, and S. Goto, “VLSI architecture of HEVC intra prediction for 8k UHDTV applications,” 2014 IEEE International Conference on Image Processing (ICIP), pp.1273–1277, October 2014.

[18] T. Ito, “Future television super hi-vision and beyond,” in Solid State Circuits

Conference (A-SSCC), 2010 IEEE Asian. IEEE, pp. 1–4, 2010

[19] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1649– 1668, 2012.

[20] "H.265: High efficiency video coding," ITU-T Rec, 2013.

[21] "Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264 — ISO/IEC 14496-10 AVC)," ITU-T Rec, 2003.

[22] C.-M. Fu, E. Alshina, A. Alshin, Y.-W. Huang, C.-Y. Chen, C.-Y. Tsai, C.-W. Hsu, S.-M. Lei, J.-H. Park, and W.-J. Han, "Sample adaptive offset in the hevc standard," IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1755–1764, 2012.

[23] P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karczewicz, "Adaptive deblocking filter," IEEE transactions on circuits and systems for video technology, vol. 13, no. 7, pp. 614–619, 2003.

[24] Y.-W. Huang, T.-W. Chen, B.-Y. Hsieh, T.-C. Wang, T.-H. Chang, and L.-G. Chen, "Architecture design for deblocking filter in h.264/jvt/avc," in Multimedia and Expo, 2003. ICME '03. Proceedings. 2003 International Conference on, vol. 1, pp. I–693–6 vol.1, July 2003.

[25] K. Xu and C.-S. Choy, "A five-stage pipeline, 204 cycles/mb, single-port sram-based deblocking filter for h. 264/avc," , IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 3, pp. 363–374, 2008.

[26] Y.-C. Lin and Y.-L. Lin, "A two-result-per-cycle deblocking filter architecture for QFHD h. 264/avc decoder," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, no. 6, pp. 838–843, 2009.

[27] D. Zhou, J. Zhou, J. Zhu, and S. Goto, "A 48cycles/mb h. 264/avc deblocking filter

architecture for ultra high definition applications,” IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences, vol. 92, pp. 3203–3210, 2009.

[28] A. Norkin, G. Bjontegaard, A. Fuldseth, M. Narroschke, M. Ikeda, K. Andersson, M. Zhou, and G. Van der Auwera, “Hevc deblocking filter,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 22, no. 12, pp. 1746–1754, 2012

[29] M. Li, J. Zhou, D. Zhou, X. Peng, and S. Goto, “A dual-mode deblocking filter design for hevc and h. 264/avc,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. 96, no. 6, pp. 1366–1375, 2013.

[30] W. Cheng, Y. Fan, Y. Lu, Y. Jin, and X. Zeng, “A high-throughput hevc deblocking filter vlsi architecture for 8k× 4k application,” 2015 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 605–608, 2015.

[31] M. Tikekar, C.-T. Huang, C. Juvekar, V. Sze, and A. P. Chandrakasan, “A 249-mpixel/s hevc video-decoder chip for 4k ultra-hd applications,” IEEE Journal of Solid-State Circuits, vol. 49, no. 1, pp. 61–72, 2014.

[32] J. Zhu, D. Zhou, G. He, and S. Goto, “A combined sao and de-blocking filter architecture for hevc video decoder,” 20th IEEE International Conference on Image Processing (ICIP), 2013, pp. 1967– 1971, 2013.

[33] M. Mody, N. Nandan, and T. Hideo, “High throughput vlsi architecture supporting hevc loop filter for ultra hdtv,” in the 3th International Conference on Consumer Electronics in Berlin (ICCE-Berlin), pp. 54–57, 2013

[34] C. M. Diniz, M. Shafique, F. V. Dalcin, S. Bampi, and J. Henkel, “A deblocking filter hardware architecture for the high efficiency video coding standard,” in Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, ser. DATE ’15. San Jose, CA, pp. 1509–1514, 2015.

[35] J. Joo, Y. Choi, and K. Lee, “Fast sample adaptive offset encoding algorithm for hevc

based on intra prediction mode," in the 3th International Conference on Consumer Electronics in Berlin (ICCE-Berlin), pp. 50–53, 2013.

[36] J. Joo and Y. Choi, "Dominant edge direction based fast parameter estimation algorithm for sample adaptive offset in hevc," in IEEE International Conference on Image Processing (ICIP), pp. 3749–3752, 2014.

[37] Y. Choi and J. Joo, "Exploration of practical hevc/h. 265 sample adaptive offset encoding policies," Signal Processing Letters, IEEE, vol. 22, no. 4, pp. 465–468, 2015.

[38] G. Chen, Z. Pei, Z. Liu, and T. Ikenaga, "Low complexity sao in hevc base on class combination, pre-decision and merge separation," in 19th International Conference on Digital Signal Processing (DSP), pp. 259–262, August 2014.

[39] F. Rediess, R. Conceicao, B. Zatt, M. Porto, and L. Agostini, "Sample adaptive offset filter hardware design for hevc encoder," in IEEE Visual Communications and Image Processing Conference, pp. 299–302, 2014.

[40] "Cost function optimization and its hardware design for the sample adaptive offset of hevc standard," in the 22nd IEEE European Proceedings of Signal Processing Conference (EUSIPCO), pp. 206–210, 2014

[41] M. Mody, H. Garud, S. Nagori, and D. K. Mandal, "High throughput vlsi architecture for hevc sao encoding for ultra hdtv," in 2014 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 2620–2623, 2014.

[42] J. Zhu, D. Zhou, S. Kimura, and S. Goto, "Fast sao estimation algorithm and its vlsi architecture," in IEEE International Conference on Image Processing (ICIP), pp. 1278–1282, 2014.

[43] J. Zhu, D. Zhou, and S. Kimura, "Fast sao estimation algorithm and its implementation for 8k× 4k@120 fps hevc encoding," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. 97, no. 12, pp. 2488–2497, 2014.

[44] F. Bossen, “Common test conditions and software reference configurations,” Joint Collaborative Team on Video Coding (JCT-VC), JCTVC- F900, 2011.

[45] S. E. Gendy, A. Shalaby and M. S. Sayed, "Fast parameter estimation algorithm for sample adaptive offset in HEVC encoder," in Visual Communications and Image Processing (VCIP), Singapore, pp. 1-4, 2015

[46] “The Internet of Things ITU Internet Rep,” 2005.

[47] G.K. Wallace and G. K., “The JPEG still picture compression standard,” Communications of the ACM, vol.34, no.4, pp.30–44, apr 1991.

[48] D. Marpe, T. Wiegand, and G. Sullivan, “The H.264/MPEG4 advanced video coding standard and its applications,” IEEE Communications Magazine, vol.44, no.8, pp.134–143, August 2006.

[49] B. Girod, A.M. Aaron, S. Rane, and D. Rebollo-Monedero, “Distributed Video Coding,” Proceedings of the IEEE 93.1,pp 71-83, 2005

[50] G.J. Sullivan, J.R. Ohm, W.J. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard,” IEEE Transactions on Circuits and Systems for Video Technology, vol.22, no.12, pp.1649–1668, December 2012.

[51] D. Donoho, “Compressed sensing,” IEEE Transactions on Information Theory, vol.52, no.4, pp.1289–1306, April 2006.

[52] R. Robucci, J.D. Gray, Leung Kin Chiu, J. Romberg, and P. Hasler, “Compressive Sensing on a CMOS Separable-Transform Image Sensor,” Proceedings of the IEEE, vol.98, no.6, pp.1089–1101, June 2010.

[53] N. Katic, M.H. Kamal, M. Kilic, A. Schmid, P. Vandergheynst, and Y. Leblebici, “Column-separated compressive sampling scheme for low power CMOS image sensors,” 11th IEEE International New Circuits and Systems Conference (NEWCAS), pp.1–4, IEEE, June 2013.

[54] Y. Oike and A. El Gamal, “CMOS Image Sensor With Per-Column ΣΔ ADC and Programmable Compressed Sensing,” IEEE Journal of Solid-State Circuits, vol.48, no.1, pp.318–328, January. 2013.

[55] M. Dadkhah, M.J. Deen, and S. Shirani, “CMOS Image Sensor With Area-Efficient Block-Based Compressive Sensing,” IEEE Sensors Journal, vol.15, no.7, pp.3699–3710, July 2015.

[56] S. Mun and J.E. Fowler, “Dpcm for quantized block-based compressed sensing of images,” Proceedings of the 20th European Signal Processing Conference (EUSIPCO), pp.1424–1428, 2012.

[57] J. Zhang, D. Zhao, and F. Jiang, “Spatially directional predictive coding for block-based compressive sensing of natural images,” 20th IEEE International Conference on Image Processing (ICIP), pp.1021–1025, Sep 2013.

[58] K.Q. Dinh, C.V. Trinh, V.A. Nguyen, Y. Park, and B. Jeon, “Measurement Coding for Compressive Sensing of Color Images,” IEIE Transactions on Smart Processing & Computing, vol.3, no.1, pp.10– 18, 2014.

[59] X. Gao, J. Zhang, W. Che, X. Fan, and D. Zhao, “Block-Based Compressive Sensing Coding of Natural Images by Local Structural Measurement Matrix,” 2015 Data Compression Conference, pp.133– 142, IEEE, April 2015.

[60] J. Zhou, D. Zhou, L. Guo, T. Yoshimura, and S. Goto, “Measurement- domain Intra Prediction Framework for Compressively Sensed Images,” 2017 IEEE International Symposium on Circuits and Systems, IEEE, May 2015.

[61] J. Zhou, D. Zhou, S. Wang, T. Yoshiumura, and S. Goto, “High Performance VLSI Architecture of H.265/HEVC Intra Prediction for 8K UHDTV Video Decoder,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, vol. E98.A, no.12, pp.2519–2527, 2015.

[62] G. Bjontegaard et al., “Improvements of the bd-psnr model,” ITU-T SG16 Q, vol.6.

[63] http://arstechnica.com/gaming/2013/09/virtual-perfection-why-8k-resolution-per-eye-isnt-enough-for-perfect-vr/

[64] M. Dadkhah, M.J. Deen, and S. Shirani, “CMOS Image Sensor With Area-Efficient Block-Based Compressive Sensing,” IEEE Sensors Journal, vol.15, no.7, pp.3699–3710, jul 2015.

[65] X.J. Liu, S.T. Xia, and T. Dai, “Deterministic constructions of binary measurement matrices with various sizes,” 2015 ICASSP, pp.3641–3645, IEEE, apr 2015.

[66] J. Zhou, D. Zhou, L. Guo, T. Yoshimura, and S. Goto, “Framework and VLSI architecture of Measurement-Domain Intra Prediction for Compressively Sensed Visual Contents,” IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, vol.E100-A, no.12, pp.2869–2877, 2017.

[67] J.Zhou,D.Zhou,L.Guo,T.Yoshimura,andS.Goto,“Approximate-DCT-Derived Measurement Matrices for Compressed Sensing,” 2017 IEEE International Symposium on Circuits and Systems, pp.1123–1126, IEEE, May 2017.

[68] F. Bayer, R. Cintra, A. Madanayake, and U. Potluri, “Multiplierless approximate 4-point DCT VLSI architectures for transform block coding,” Electronics Letters, vol.49, no.24, pp.1532–1534, 2013.

[69] U.S. Potluri, A. Madanayake, R.J. Cintra, F.M. Bayer, S. Kulasekera, and A. Edirisuriya, “Improved 8-Point Approximate DCT for Image and Video Compression Requiring Only 14 Additions,” IEEE TCAS-I: Regular Papers, vol.61, no.6, pp.1727–1740, jun 2014.

[70] T.L.T. da Silveira, F.M. Bayer, R.J. Cintra, S. Kulasekera, A. Madanayake, and A.J. Kozakevicius, “An orthogonal 16-point approximate DCT for image and video compression,” Multidimensional Systems and Signal Processing, vol.27, no.1, pp.87–104, jan 2016.

[71] N. Katic, M.H. Kamal, M. Kilic, A. Schmid, P. Vandergheynst, and Y. Leblebici,“Power-efficient CMOS image acquisition system based on compressive sampling,” 2013 IEEE 56th MWSCAS, pp.1367–1370, IEEE, aug 2013

[72] X. He, D. Zhou, J. Zhou, and S. Goto, “High profile intra prediction architecture for UHD H.264 decoder”, IPSJ Transactions on System LSI Design Methodology, Vol. 3, No. 2, pp. 303-313, August, 2010.

Publication List

The papers marked with circles are presented in the dissertation.

Journal Paper (with review):

○ [1] Jianbin Zhou, Dajiang Zhou, Takeshi Yoshimura and Satoshi Goto “Approximate-DCT-Derived Measurement Matrices with Row-Operation-Based Measurement Compression and its VLSI Architecture for Compressed Sensing”, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol.E101-C, No.4, pp.263-272, Apr. 2018.

○ [2] Jianbin Zhou, Dajiang Zhou, Li Guo, Takeshi Yoshimura and Satoshi Goto “Framework and VLSI Architecture of Measurement-Domain Intra Prediction for Compressively Sensed Visual Contents”, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol.E100-A, No.12, pp.2869-2877, Dec. 2017.

○ [3] Jianbin Zhou, Dajiang Zhou, Shihao Wang, Shuping Zhang, Takeshi Yoshimura and Satoshi Goto, "A Dual-Clock VLSI Design of H.265 Sample Adaptive Offset Estimation for 8k Ultra-HD TV Encoding," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 25, no. 2, pp. 714-724, Feb. 2017

[4] Dajiang Zhou, Shihao Wang, Heming Sun, Jianbin Zhou, Jiayi Zhu, Yijin Zhao, Jinjia Zhou, Shuping Zhang, Shinji Kimura, Takeshi Yoshimura, and Satoshi Goto; "An 8K H.265/HEVC Video Decoder Chip With a New System Pipeline Design," in IEEE Journal of Solid-State Circuits (JSSC), vol. 52, no. 1, pp. 113-126, Jan. 2017.

[5] Shihao Wang, Dajiang Zhou, Jianbin Zhou, Takeshi Yoshimura, and Satoshi Goto, VLSI implementation of HEVC motion compensation with distance biased direct cache mapping for 8K UHDTV applications, IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), vol. 27, no. 2, pp. 380-393, Feb. 2017.

○ [6] Jianbin Zhou, Dajiang Zhou, Shihao Wang, Takeshi Yoshimura and Satoshi Goto, “High performance VLSI architecture of H. 265/HEVC intra prediction for 8K UHDTV video decoder", IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E98.A (2015) No. 12 pp. 2519-2527, Dec. 2015.

Publication List

International Conference Paper (with review):

○ [1] Jianbin Zhou, Dajiang Zhou, Li Guo, Takeshi Yoshimura and Satoshi Goto; “The Measurement-Domain Intra Prediction Framework for Compressively Sensed Images”. IEEE International Symposium on Circuits & Systems (ISCAS), pp.168-171 May 2017.

○ [2] Jianbin Zhou, Dajiang Zhou, Takeshi Yoshimura and Satoshi Goto; “Approximate-DCT-Derived Measurement Matrices for Compressed Sensing,” IEEE International Symposium on Circuits & Systems (ISCAS), pp. 1123-1126 May 2017.

[3] Kaiyi Yang, Shihao Wang, Jianbin Zhou, Takeshi Yoshimura, “Energy-efficient Scheduling Method with Cross-loop Model for customized CNN Accelerators,” IEEE International Symposium on Circuits & Systems (ISCAS), pp. 2046-2049, May 2017.

[4] Jianbin Zhou, Dajiang Zhou, Shihao Wang, Shuping Zhang, Takeshi Yoshimura and Satoshi Goto, "A Dual-Clock VLSI Design of H.265 Sample Adaptive Offset Estimation for 8k Ultra-HD TV Encoding," IEEE International Symposium on Circuits & Systems (ISCAS), pp. 2046-2049, May 2017.(Invited talk)

[5] Dajiang Zhou, Shihao Wang, Heming Sun, Jianbin Zhou, Jiayi Zhu, Yijin Zhao, Jinjia Zhou, Shuping Zhang, Shinji Kimura, Takeshi Yoshimura, and Satoshi Goto; “14.7 A 4Gpixel/s 8/10b H. 265/HEVC video decoder chip for 8K Ultra HD applications ” 2016 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, 2016, pp. 266-268, Feb. 2016.

[6] Shihao Wang, Dajiang Zhou, Jianbin Zhou, Takeshi Yoshimura and Satoshi Goto,“Unified VLSI Architecture of Motion Vector and Boundary Strength Parameter Decoder for 8K UHDTV HEVC Decoder,” Pacific Rim Conference on Multimedia (PCM), p. 74-83, December, 2014.

○ [7] Jianbin Zhou, Dajiang Zhou, Heming Sun, Satoshi Goto, “VLSI architecture of HEVC intra prediction for 8K UHDTV applications," IEEE International Conference on Image Processing (ICIP), 2014, vol., no., pp.1273, 1277, 27-30 Oct. 2014.