ERROR TOLERANT MULTIMEDIA COMPRESSION SYSTEM

by

In Suk Chong

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

December 2009

Copyright 2009 In Suk Chong

Dedication

To my parents, Young-Do and Gui-Sim, parents-in-law, Young-Shik and Hyung-Ae, and my wife Young Eun.

Acknowledgements

First of all, I would like to express my sincere gratitude to my advisor, Prof. Antonio Ortega who has supported me throughout my doctoral research with his enthusiasm, advice, and great patience. Without his mentorship, this dissertation would not be done.

I would like to thank Prof. Sandeep K. Gupta and Prof. Ming Deh Huang for serving on my dissertation committee, and Prof. Keith M. Chugg and C. C. Jay Kuo for their guidance and serving on my Ph.D. qualification exam committee. I also would like to thank Prof. Melvin A. Breuer for his advice and sharing ideas during my research on the error tolerant computing.

My deepest thanks also go to many student colleagues in the Compression Research Group, former members Ngai-Man Cheung, Hyukjune Chung, Alexandre Ciancio, Jae Hoon Kim, Polin Lai, So Yeon Lee, Huisheng Wang, and Hua Xie; and current members, Hye-Yeon Cheong, Woo-Shik Kim, Kun-Han Lee, Sungwon Lee, Sean McPherson, Godwin Shen, and Ivy Tseng. I will never forget the time we have had for a interesting discussion and fun.

I would like to thank my family members, specially my beloved parents, parents-in-law, and my brothers, for their love and continuous prayers for me. Especially, I would like to give my thanks to my wife, Young Eun, for putting her faith in me and for giving me her love and sacrifice through all these years. Lastly, and most

importantly, I would like to say that the whole things were, are, and will be under the guidance of God.

Table of Contents

Dedication ii

Acknowledgements iii

List of Tables vii

List of Figures viii

Abstract x

Chapter 1 Introduction 1
- 1.1 Error Tolerance in Multimedia Compression 1
- 1.2 Error Tolerance in Motion Estimation 5
- 1.3 Low Complexity Motion Estimation 8
- 1.4 Error Tolerance in DCT 9

Chapter 2 Low Complexity ME using Voltage Over Scaled MMC 11
- 2.1 Introduction 11
- 2.2 Motion Estimation with Soft Error 13
  - 2.2.1 Errors within a voltage over-scaled adder 15
  - 2.2.2 The Soft Error Model 17
- 2.3 Modeling the effect of soft errors on the coding performance 19
- 2.4 Simulation and Discussion 22
- 2.5 Conclusion 25

Chapter 3 Low Complexity ME using Multiple Imprecise MMC 28
- 3.1 Introduction 28
  - 3.1.1 Previous work 28
  - 3.1.2 Contribution 30
- 3.2 Framework for multiple matching metric Motion Estimation 33

3.3 Desirable characteristics of multiple metrics and CD curve estimation 36
3.4 MV estimator 39
3.4.1 Hypothesis testing 39
3.4.2 Sub-optimal MV estimator 41
3.5 Case study: MV estimators using VOS and SS metrics 41
3.5.1 Characteristics of VOS and SS errors 42
3.5.2 Threshold estimator 43
3.5.3 MAX estimator 44
3.5.4 MV estimator using hypothesis testing 45
3.6 Case study: Desirable characteristics of multiple metrics and CD curve estimation 46
3.7 Simulation Results 49
3.8 Performance under different ME algorithms and MMC architectures 51
3.9 Conclusion 53

Chapter 4 Error Tolerant DCT 55
4.1 Introduction 55
4.2 System level error tolerance in DCT and quantization 58
4.2.1 DCT and Quantization 60
4.2.2 Quantization Analysis 62
4.2.3 Analysis of additional error due to fault: ΔD 64
4.3 Metric and threshold 68
4.3.1 Metric Selection for Block-wise Error Significance 70
4.3.2 Block-wise error significance and error rate 72
4.4 Fault analysis 74
4.5 Systematic Test Vector generation for DCT 76
4.5.1 ER-ES plot and Generation of test vectors 78
4.5.2 Comparison of Testing qualities and costs 81
4.5.3 Increases in Yield Rate 83
4.6 Conclusion 85

Chapter 5 Conclusion and Future Work 86

Bibliography 88

Appendix Advantages of positive biased estimator 93

List of Tables

1.1 Error tolerance of an video encoder in Fig. 1.3 [17] 5

4.1 Changes in decoded image quality due to a SSF at the input of DCT. The quality change is measured for a test image by computing the difference in the PSNR of the decoded image obtained from a fault-free JPEG encoder and that produced by a faulty JPEG encoder. A negative value corresponds quality degradation. Bit 0 corresponds to the least significant bit of an input pixel. The parameter R controls the quantization level, with larger R corresponding to coarser quantization (i.e., lower bit-rate and worse decoded image quality.) 57

4.2 Possible error rate threshold (R_{th}) values depending on the resolution values, where $N_{8\times8}$ is the number of 8×8 blocks within one image or frame 74

4.3 Original yield rate (Y_r) vs. new yield rate (Y_r^{new}) in % due to tests considering error tolerance for the DCT system with a non-sequential PE; $Q = 1$ and $R_{th} = 0.0001$ 84

List of Figures

1.1 Error tolerance 2

1.2 Stuck At Faults 4

1.3 General video encoder/decoder block diagram 4

1.4 Motion estimation schematic diagram 7

2.1 MMC Architectures. (a): Serial Architecture, (b): Parallel Architecture 14

2.2 Upper left: input with R_I equal to 6, Lower left: output S with R_I equal to $(K+1)T_{FA}$, Right: soft error due to Vdd control 16

2.3 Rate change due to DVS in ratio compared to original rate (dot: estimated data using our model, solid: real data); using FOREMAN sequence, QP=20 26

2.4 Power saving effect of ME process using DVS for various ME algorithms and MMC architectures; considering redundancy due to encoding one frame of every GOP with Vdd_{crit} and R_S^L for SAD_i information and X_1 estimation, FOREMAN sequence, QP=20, gate parameters $(\alpha = 2.0, Vdd_{crit} = 3.3V, V_t = 0.62V)$ 27

3.1 ME with two imprecise metric computations; SAD is used as matching metric cost 30

3.2 MMC architectures [33,50]. (a) Structure of AD and serial architecture with one AD, (b) Serial architecture with M^2 ADs, (c) Parallel architecture with M^2 ADs 33

3.3 Frameworks for MV estimation: (a) hypothesis testing using likelihoods jointly, (b) estimation of each SAD_i followed by minimum operation 35

3.4 Two SSVOS metrics combined with adder and MAX 48

3.5 Estimation of CD curves 48

3.6 Comparison of MV estimators and metric combinations, FOREMAN sequence, gate parameters ($\alpha = 1.2, Vdd_{crit} = 3.3V, V_t = 0.62V$) 51

3.7 Comparison ME algorithms and MMC architectures with $m = 4$, FOREMAN sequence, gate parameters ($\alpha = 1.2, Vdd_{crit} = 3.3V, V_t = 0.62V$) 54

4.1 DCT block diagram 59

4.2 PE architectures; (a) non-sequential, (b) sequential 59

4.3 DCT and Quantization 61

4.4 Quantization Analysis 62

4.5 Quantization Analysis. Y is quantized into bins of size Δ. If we focus on one bin (I_l), $Q(Y) = l\Delta$ is the center of that bin. $Y' = Y + e$ (where $e = L\Delta + e'$) now belongs to a different quantization bin. There are two cases for a given error e. If $Y \in I_l^1$ then $Q(Y') = (l + L + 1)\Delta$. Alternatively, $Y \in I_l^0$ leads to $Q(Y') = (l + L)\Delta$. . . . 63

4.6 Relation between ΔD and Y for (a): $L = 0$ and (b): $L \neq 0$. At the edge of I_l^1 ΔD has maximum. $\mathbf{B_l}$ is the range of Y in each quantization bin (I_l) where ΔD is larger than error threshold (E_{th}), and $\mathbf{C_l}$ is the range of Y in each quantization bin (I_l) where ΔD is smaller than E_{th}. We need to integrate $f_Y(y)$ over this range to get error rate. 65

4.7 (a): Higher MSE but Acceptable, (b): Lower MSE but unacceptable 68

4.8 Fault analysis results 74

4.9 ER-ES plot 79

4.10 How to decide ER and ES threshold values 79

4.11 p_k computation; dotted: p_0, solid: p_1 81

4.12 The percentage of acceptable faults using Ideal, ES, and ER&ES tests with both uniform (U) and non-uniform (NU) input sets; for $R_{th} = 0.0001$ and a non-sequential PE 82

Abstract

All existing digital system design approaches strive to provide error-free values at system outputs, which is achieved by using testing techniques, as well as defect and fault tolerant methods. In contrast, the focus of this thesis is on error tolerant systems, i.e., systems where certain types of errors at system outputs can be tolerated, provided their severities are below certain levels. The overall objective of our research in error tolerance is (i) to identify the inherent ability of some systems to tolerate errors, and (ii) to develop design and test approaches that exploit this ability.

Our target systems are multimedia applications, which represent a major workload on major hand-held devices such as cellular phones and laptops. Video coders (e.g., H.264/AVC [3] and MPEG-4 [1]) and image coders (e.g., JPEG [36] and JPEG2000 [2]) are the most complex part of multimedia applications. Within a typical video/image coder, we focus on motion estimation (ME) and linear transforms (e.g., discrete cosine transform) as those two consume a large percentage of resources. Achieving error tolerance in those two modules can lead to increased yield or lower power consumption.

In this thesis, we study two specific scenarios i) soft-error tolerance in matching metric computations within motion estimation, ii) hard-error tolerance in linear

transforms (e.g., discrete cosine transform). While soft errors can also be introduced due to deep submicron (DSM) noise, we focus on voltage over scaling which introduces input-dependent errors. We show that soft errors within matching metric computation of motion estimation are tolerable to some extent. The tolerance to soft errors of the motion estimation module is exploited in a low power motion estimation system; i.e., the motion estimation module can operate (with some error) at a lower voltage configuration, thus achieving power savings. We first explore one possible configuration which uses one voltage over scaled metric computation within ME. We then extend that work to a more general configuration using multiple low complexity metric computations (e.g., voltage over scaled and sub-sampled metrics) within ME.

Hard errors are introduced due to defects within hardware. By emulating the effect of those hard errors within linear transform hardware, we show that linear transforms have significant error tolerance. This error tolerance is exploited in order to achieve a higher yield rate by accepting systems with faults, while still operate with acceptable quality. To realize this, we introduce a systematic error tolerant testing method for this hardware.

Chapter 1

Introduction

1.1 Error Tolerance in Multimedia Compression

The progress of VLSI technology towards deep sub-micron feature sizes, e.g., sub-100 nanometer technologies, is resulting in a growing impact of hardware defects and fabrication process variability. This research is motivated by two scenarios where this trend leads to imperfect hardware systems, namely, soft and hard errors. For both cases we study the computation error tolerance (ET) [8] properties of specific hardware applications, which allows us to accept systems exhibiting errors at their outputs, as long as these result in only "acceptable" degradation in performance, e.g., in terms of coding efficiency. Determining what constitutes acceptable degradation in system performance is obviously an application-specific decision; both performance criteria and acceptability thresholds are highly dependent on the application. Even though this is an application-specific decision, we address this problem in a general framework by providing systematic methods to characterize and quantitatively measure the impact of those errors and to determine the thresholds.

Hardware defects lead to faults at circuit interconnects, which can potentially lead to "hard" errors, since some of the functionality in the design is permanently impaired. Traditionally, systems having these kinds of faults would be discarded after testing. Defect tolerance techniques at the design and manufacturing stages have been widely studied and used in practice [32]. However, our focus is on system-level error tolerance. After appropriate testing, any imperfect system that is deemed as providing acceptable quality can be considered usable, thus increasing the effective yield rate of the fabrication process.

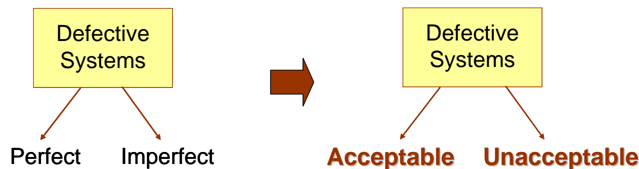


Fig. 1.1: Error tolerance

Soft errors are introduced due to deep submicron (DSM) noise and voltage over scaling which cause probabilistic and input-dependent errors in the systems [28,45]. We focus on soft errors due to voltage over scaling, which may arise when a circuit operates at a voltage lower than originally specified. Due to variability in the fabrication process, each version of a fabricated system may require a slightly different voltage to guarantee error-free operation. Our motivation in exploring error tolerance in this context is that, for certain applications, such as video coding, these systems may be able to operate at voltages below those originally specified, leading to lower power consumption (and thus longer battery life) [23]. An important part of our work is to design systems where acceptable quality can be achieved, while both hardware related soft errors and algorithmic errors (e.g., due to algorithmic simplifications to speed up processing) coexists.

In this thesis, we consider multimedia compression systems as a promising application area for our proposed ET concepts. This is because i) many multimedia compression systems are deployed in consumer devices, for which maintaining low costs and/or low power is important, and ii) compression itself leads to a lossy representation of signals, so that the effect of system faults can be viewed as an additional source of "noise" or representation error.

Within multimedia compression systems, we focus on two very common components, namely the discrete cosine transform (DCT) and motion estimation (ME), which are known to be error tolerant within MPEG and JPEG systems [18] (see Tab. 1.1). Note that those two modules consume a significant share of resources within a typical multimedia compression systems (e.g., 66%-94% in an MPEG-4 encoder [33]). The DCT is used in video coders, e.g., MPEG [37], and image coders, e.g., JPEG [36], and similar linear block transforms are used in more recent compression systems, such as ITU-T H.264 [3]. Note that in all these systems the transform is followed by quantization. Thus, while we consider faults in the transform operation, our analysis considers the impact of the faults *after* quantization. In this thesis, we consider faults in the interconnect data bus (edge) within DCT. We model interconnect faults with the single stuck-at (SSA) fault model (see Fig. 1.2), a well-known structural fault model which assumes that the design contains a fault that will cause a line in the circuit to behave as if it is permanently stuck at a logic value 0 (stuck-at 0) or 1 (stuck-at 1). The SSA fault model covers 80-90% of the possible manufacturing defects in CMOS circuits [47], such as missing features, source-drain shorts, diffusion contaminants, and metallization shorts, oxide pinholes, etc.

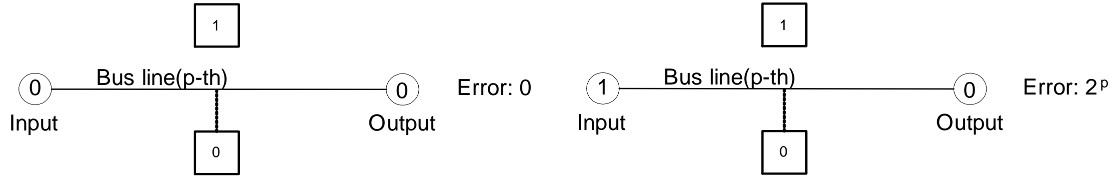


Fig. 1.2: Stuck At Faults

In case of the motion estimation module, we consider soft errors in the voltage over scaled (VOS) adders within a matching metric computation (MMC) architecture. We model the errors produced by VOS on ripple carry adders. Then for the ripple carry adder case, we propose low power motion estimation algorithm based on the model of VOS errors. Note that similar error models can be derived for other type of adders, thus our results can be easily extended to other adders. Based on this work, we explore techniques to combine *multiple low complexity* metrics (e.g., a combination of voltage over scaled and sub sampled metrics) to achieve a low complexity (or power) ME.

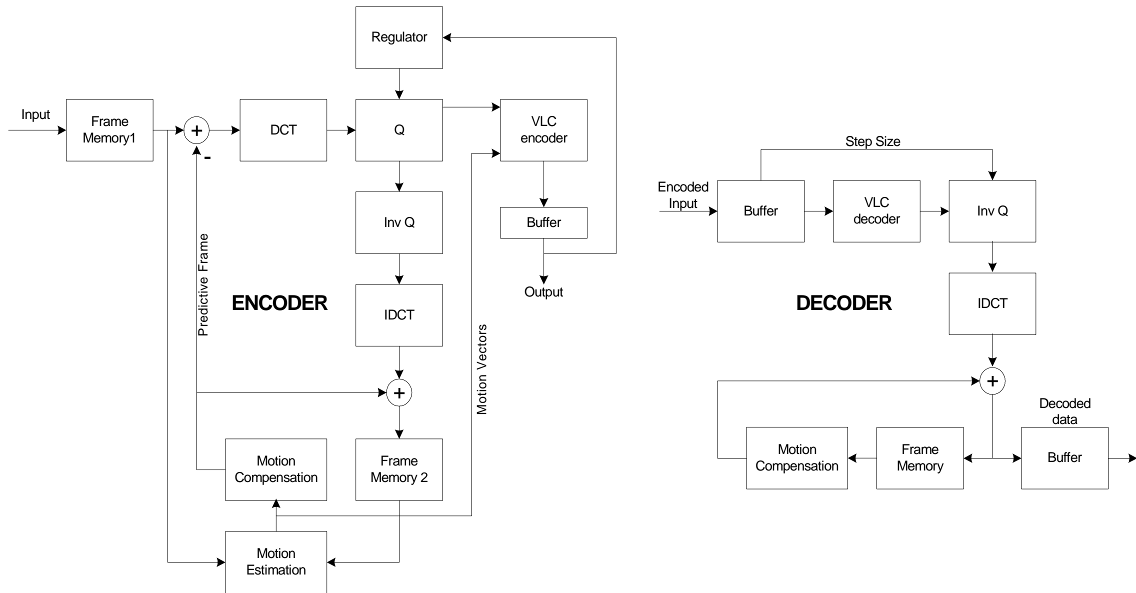


Fig. 1.3: General video encoder/decoder block diagram

	Catastrophic Error	Error Propagation	Quality Degradation	Coding Efficiency Degradation
Frame Memory	No	No	Yes	Yes
DCT	No	No	Yes	Yes
Q	No	No	Yes	Yes
Inv Q	No	Yes	Yes	Yes
IDCT	No	Yes	Yes	Yes
Frame Memory	No	Yes	Yes	Yes
ME	No	No	Possible	Yes
MC	No	Yes	Yes	Yes
VLC Encoder	Yes	Yes	Yes	Yes

Tab. 1.1: Error tolerance of an video encoder in Fig. 1.3 [17]

Note that we are not considering fault tolerance techniques which aim at achieving the same performance with a faulty system as would be achieved with a fault-free system. Instead, we consider systems that produce errors at their outputs (e.g., the motion vectors of the faulty system can be different from those of the fault-free one) and evaluate the impact of these errors on overall performance (e.g., coding efficiency).

1.2 Error Tolerance in Motion Estimation

Motion estimation is an efficient tool for video compression which allows exploiting temporal correlation between adjacent frames in video sequence. In this research, we will use block matching, which is the most widely used motion estimation, compensation technique.

A motion estimation algorithm is composed of a searching process and a matching process. The matching process is the process which computes the distortion between the current block and a candidate block. The searching process selects

candidate blocks from given possible set of candidates. The most straightforward but computationally expensive search algorithm is the full search (or exhaustive search) algorithm. For block matching motion estimation using full search, a block of size $M \times M$ (reference macro block X) of the current image is matched with all the blocks in the search window of size $(2w+1) \times (2w+1)$ (i-th candidate blocks whose displacement is (p, q) is denoted Y_i). The motion estimation can be described as,

$$SAD_i = \sum_{(l,m) \in A} |x(l,m) - y_i(l,m)|, i = 1, 2, ..., N \tag{1.1}$$

$$MV_{min} = \arg\min_i SAD_i \tag{1.2}$$

where A is $[0, M-1] \times [0, M-1]$, $x(l,m)$ and $y_i(l,m)$ are the (l,m)-th pixel values of blocks X and Y_i, respectively. The output from a motion estimation is a motion vector MV_{min} which indicates the translational motion between a current block and the best candidate block from a search. In the matching process, typically the sum of absolute differences (SAD) or the sum of squared differences (SSD) are used for metric, and SAD is chosen in this work (see Fig. 1.4).

In [11,17], it is shown that the MMC process within ME is error tolerant, i.e., certain range of deterministic faults and soft errors within MMC lead to acceptable quality degradation, while both video encoder and decoder remain operational. When the MMC process is affected by soft/hard errors, SAD_i values may be corrupted; those possibly erroneous SAD_i values are denoted SAD'_i. Denote MV_f the MV chosen when the SAD'_i are used in (1.2). If $MV_f \neq MV_{min}$, the residual block's distortion (as measured by the SAD) increases by $E_{SAD} \triangleq SAD_{MV_f} - SAD_{min}$.

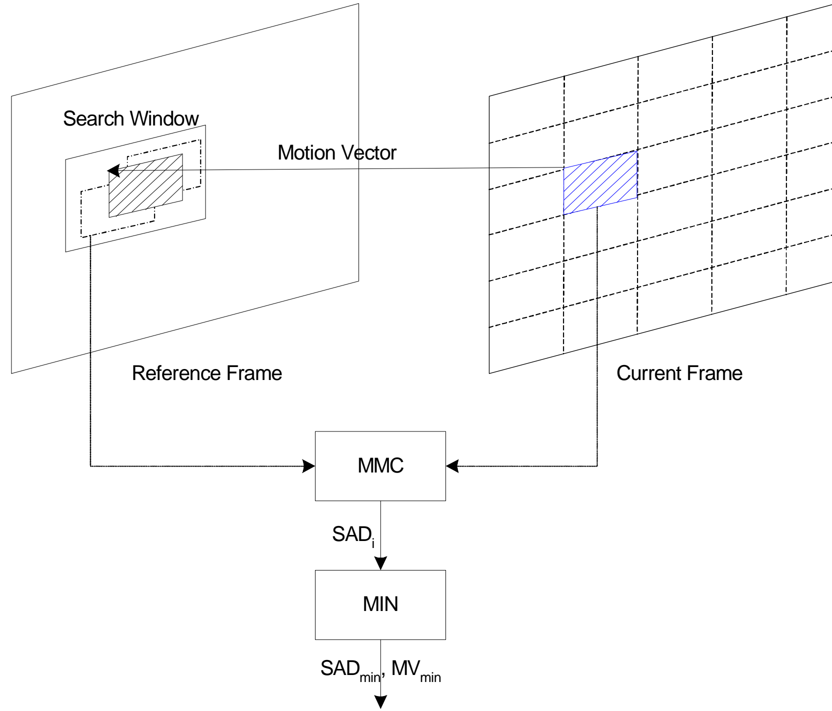


Fig. 1.4: Motion estimation schematic diagram

This increase in distortion (E_{SAD}) may lead to degradation in compression performance, and should be evaluated in terms of this compression performance penalty (e.g., more bits may be needed to code data at a given quality level, as compared to the bit-rate required by a fault-free system operating at the same QP). This performance degradation may be acceptable for specific application scenarios. In [11], a primarily *experimental* evaluation of the behavior of several ME algorithms are provided when soft errors are applied to MMC within ME process in the context of H.264/AVC [3]. Note that this error tolerance is general and not restricted to a specific search algorithm or architecture, even though the degree of the tolerance depends on them.

1.3 Low Complexity Motion Estimation

Algorithmic approaches for power efficient ME [21, 22, 48] have been studied for a number of years. Recently, a new technique, voltage over scaling within ME, has been shown to lead to significant additional power savings [15, 49] when the input voltage (Vdd) for the SAD computation module is set below critical voltage (Vdd_{crit}). A reduction in Vdd by a factor W can lead to power dissipation that is nearly a factor of W^2 lower. A major difference with respect to existing algorithmic methods is that the lower power consumption comes at the cost of *input-dependent soft errors*; lower input voltage increases circuit delay, and the number of basic operations possible for one clock period decreases, thus generating error (see [28] for an introduction to the soft DSP approach).

In this thesis, we apply VOS to the MMC used within the ME of typical video encoders. Our approach is based on soft DSP concepts. We propose an *analytical* model for the the effect of ME errors (due to VOS) in the overall coding performance as a function of input characteristics and input voltage, for given ME algorithm and MMC architecture. Note that this model is derived by characterizing the behavior of voltage over-scaled basic arithmetic operations (i.e., addition), so that it can be easily extended to scenarios which adopt voltage over-scaled operations. This model is validated using simulations. We then compare ME algorithms and MMC architectures, and propose a method for power saving of the ME process that depend on input characteristics and desired coding performance. As an illustration of the potential benefits of allowing computation errors, we show that allowing errors that lead to a small rate increase (about 3%) produces 37% power

savings in the ME process, as compared to not using VOS. An essentially "error-free" VOS approach (no rate penalty) can achieve around 10% power savings. Note that these savings may depend on the circuit parameters, i.e., V_t, Vdd_{crit}, α.

We also extend the above work by providing a general framework for multiple matching metric ME, which considers the trade off between complexity and ME system performance degradation. Within this framework, we discuss (i) desirable characteristics for these metrics, (ii) how to analyze the complexity-degradation trade off without extensive simulation, and (iii) how to combine multiple metric computations to estimate the best motion vector (MV). Furthermore, we compare various configurations, by using different ME algorithms and MMC architectures. As a case study, we apply the above framework to the combination of VOS and sub-sampled (SS) metrics. We introduce a new two-metric system whose performance is better than the combination of VOS and SS metrics (87% power saving with less than 2% degradation).

1.4 Error Tolerance in DCT

In this thesis, we propose a system-level error tolerance scheme for systems where a linear transform is combined with quantization. Using the concept of acceptable degradation, our scheme classifies hardware faults into acceptable and unacceptable faults. We propose analysis techniques that allow us to estimate the faults' impact on compression performance, and in particular on the quality of decoded images/video. We consider as an example the DCT, which is part of a large number of existing image and video compression systems. We propose methods to establish thresholds of acceptable degradation and corresponding testing algorithms for

DCT-based systems. Our results for a JPEG encoder using a typical DCT architecture show that over 50% of single stuck-at interconnection faults in the DCT modules lead to imperceptible quality degradation in the decoded images, over the complete range of compression rates at which JPEG can operate. We also propose a systematic testing method which uses a composite of error rate and error significance metrics for the DCT submodules. This new testing methods increase the yield rate $1.4 \sim 2.0$ times for low to mid range original yield rate. For example, for 20% original yield rate, improved yield rate is 40.8%.

In summary, we propose a system-level error tolerance scheme for the key components in multimedia compression systems, i) a linear transform (e.g., DCT) combined with quantization and ii) motion estimation. This error tolerance provides i) yield rate increase for the DCT system of around factor of $1.4 \sim 2.0$, for low to mid range original yield rate, and ii) savings in power for the ME system, e.g., 87% power saving with negligible performance degradation. The rest of the thesis is organized as follows. In Chapter 2, we propose low power motion estimation process using voltage over scaled matching metric computation within motion estimation in typical video encoders. In Chapter 3, we propose power efficient motion estimation using multiple imprecise matching metric computations. In Chapter 4, we propose a system-level error tolerance scheme for systems where a linear transform is combined with quantization. Concluding remarks and future research directions are given in Chapter 5.

Chapter 2

Low Complexity ME using Voltage Over Scaled MMC

2.1 Introduction

Power (or energy) is the most important design constraint in many VLSI design scenarios [38]. Many approaches have been proposed for power constrained VLSI, ranging from circuit level to architectural and algorithmic level [19, 28]. Dynamic voltage scaling (DVS) is an attractive technique to reduce power consumption, as lowering input voltage by a factor of J, reduces energy dissipation by almost a factor of J^2 [28]. Soft DSP is an example of DVS [28] that has been applied to low level systems, such as adders and multiplier-accumulators (MACs) often used in signal processing applications (e.g., linear filters and multi-input-multi-output, MIMO, systems). In soft DSP systems the input voltage can be set below critical voltage (and thus, we would have voltage over scaling, VOS), which leads to input-dependent soft errors. Then, soft-error tolerance is achieved by using explicit error

control blocks that provide error concealment in order to operate with negligible loss in algorithm performance.

In [10, 14, 17, 17], it is shown that image/video compression systems exhibit ET characteristics, *even if no explicit error control block is added*, and this is observed under both hard errors (due to deterministic faults) and soft errors (due to DVS). Errors due to VOS in these applications are either i) concealed by other parts of the system (e.g., quantization can conceal errors affecting a transform computation) or ii) are "acceptable" [8]. Determining what constitutes acceptable errors is obviously an application-specific decision; both performance criteria and acceptability thresholds are highly dependent on the application. In [17], the impact of hard errors on MMC within ME process is studied, i.e., both video encoder and decoder remain operational, and thus these errors can be evaluated in terms of the compression performance penalty they produce (i.e., more bits are needed to code data at a given quality level as compared to the bit-rate required by a fault-free system operating at the same QP). This performance penalty may be acceptable for specific application scenarios. Also a primarily *experimental* evaluation of the behavior of several ME algorithms under "soft error" conditions applying soft DSP approaches to MMC within ME process, is proposed in [10].

In this thesis, we extend the work in [10] to a DVS scenario. The main novelty comes from i) a model for degradation in video coding performance due to voltage scaling, as a function of input characteristics and for given ME algorithms and MMC architectures (this model can used to select input voltage values for target coding performance criteria), and ii) using this model to compare various ME algorithms and MMC architectures regard to their coding performance under DVS. To acquire this model, we first characterize the behavior of low level errors (i.e.,

errors in voltage over-scaled adders within MMC architectures) and then show how those low level errors lead to high level errors (i.e., video coding performance degradation). This type of analysis has not been considered in literatures yet. Our proposed models for DVS performance are designed to be used in hardware-based video encoders, but could also be useful in the context of general purpose processors for which power control is enabled (see [23] for an example). Since ME is performed at the encoder, our work is primarily applicable to scenarios where power-constrained devices (e.g., cellphones) are used for video capture and encoding.

To introduce our model, we first briefly explain the ME process and introduce different MMC architectures we will use. Each MMC architecture involves several "soft" adders, such as those used in the soft DSP context. We provide a detailed analysis of soft errors due to voltage scaling for a single adder and MMC architectures (Section 2.2). Then we extend it to model the performance degradation caused by the soft error as a function of input voltage and input characteristics (Section 2.3). This model is validated using simulations (Section 2.4). Using this model, we propose a simple voltage control method, which can achieve about 37% power savings in the ME process, as compared to not applying any voltage scaling, with very slight increase in rate (around 3%). This simple approach will be extended to more sophisticated approaches in Chapter 3.

2.2 Motion Estimation with Soft Error

The ME process comprises a search strategy and a matching process where a matching metric computation is involved, as described in Section 1.2. The search

strategy identifies a set of candidate motion vectors (MVs) and then proceeds to compute the matching metric for the candidates and to select the one that minimizes the matching metric (e.g., SAD). It is important to consider matching metric computation architectures because different architectures have different errors when those are operating in voltage over scaled conditions. There are several types of hardware architectures to compute the matching metrics, with different levels of parallelism. We will refer to them as MMC architectures. Among those, we consider both serial and parallel architectures, where M^2 basic operations (i.e., absolute difference computations and additions) are performed to compute SAD between two $M \times M$ macro-blocks (see Fig. 2.1) [39].

The serial architecture in Fig. 2.1 (a) has M^2 serially connected adders, while the parallel architecture in Fig. 2.1 (b) has M parallel groups of "leaf" adders and M "central" adders. Each group of leaf adders consists of $M-1$ adders and computes the sum of M absolute difference (AD) values. Then, the central adders compute the final SAD adding up M partial SAD values.

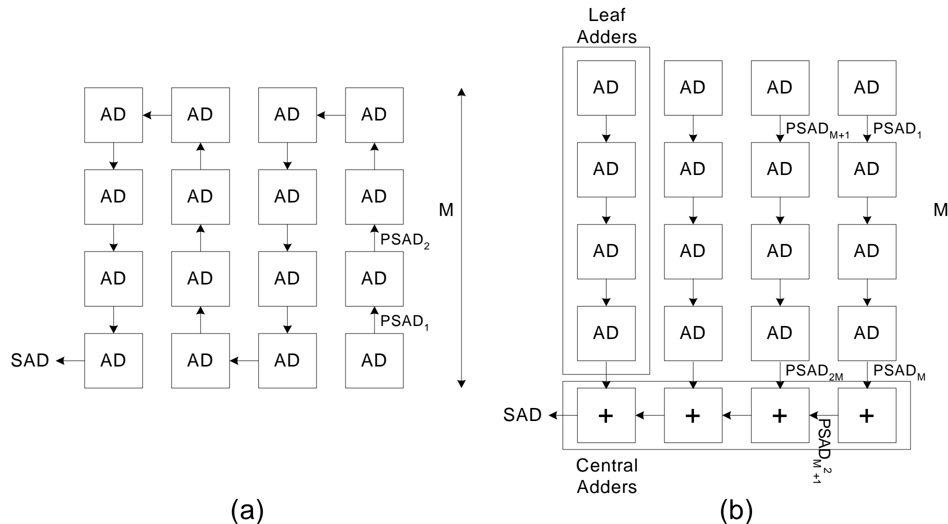


Fig. 2.1: MMC Architectures. (a): Serial Architecture, (b): Parallel Architecture

In this thesis, our focus is on voltage over scaled adders within MMC. Now we will consider the behavior of those adders.

2.2.1 Errors within a voltage over-scaled adder

An SAD computation includes several n-bit adders. We assume that voltage over-scaled ripple carry adders are used, as they provide useful functionality for VOS metric [28]. Note that similar error models can be derived for other types of adders, thus our results can be easily extended to other adders. Those adders have the same input voltage (Vdd) and fixed sampling period (T_S), as is typically assumed in soft DSP techniques [28]. An n-bit ripple carry adder comprises n serially connected full adders. When we decrease Vdd, the time needed for one full adder operation (T_{FA}) increases. Consequently, R_S, the number of full adder (FA) operations possible in T_S, decreases. If the number of FA operations required to complete one addition (say R_I, which is obviously input dependent) is larger than R_S, then an error will be generated. For example, for $T_S = 1600ns$, $T_{FA}^1 = 100ns$ for Vdd_1, and $T_{FA}^2 = 200ns$ for $Vdd_2 < Vdd_1$, translates into $R_S^1 = 16$ and $R_S^2 = 8$. For an input requiring 10 FA operations (i.e., $R_I = 10$), Vdd_2 generates error while Vdd_1 does not.

To understand the characteristics of VOS errors in detail, we look into the behavior of an n-bit soft adder (n-bit ripple carry adder with voltage over-scaling). Denote its inputs $A = [a_n...a_1]$ and $B = [b_n...b_1]$, and let $S = A + B = [s_n...s_1]$, with the carry denoted by $C = [c_n...c_1]$. Each FA has inputs a_i, b_i, c_{i-1} and outputs s_i, c_i, and each input pair (a_i, b_i) is introduced to corresponding FA simultaneously. If a carry c_i is generated in each FA, it is propagated to the next FA (see Fig. 2.2).

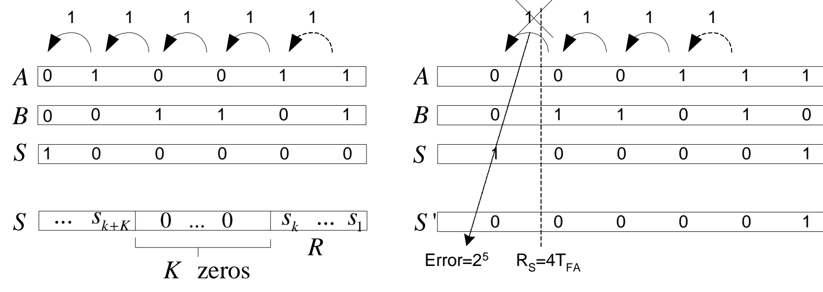


Fig. 2.2: Upper left: input with R_I equal to 6, Lower left: output S with R_I equal to $(K+1)T_{FA}$, Right: soft error due to Vdd control

R_I is determined by the *longest consecutive carry propagations*. Consecutive carry propagations are generated after an initial carry generation (by an $(1,1)$ input pair) which is followed by carry propagation inputs ($(1,0)$ or $(0,1)$ input pairs). If an input has an R_I larger than R_S, consecutive carry propagation chain will be *broken* and error is generated due to *carry loss*. Note that the position of carry loss is R_S bits away from initial carry position (i.e., $k-th$), which results in an error with magnitude -2^{R_S+k} (see Fig. 2.2). We observe the following facts: i) an error cannot be generated if output of adder is less than 2^{R_S}, whose maximum number of consecutive carry propagation is always less than R_S, ii) errors are non-positive with magnitude -2^{R_S+k}, and iii) faulty result is always non-negative, because our adder operates on non-negative inputs, which means that in turn the magnitude of errors can be no greater than fault-free result.

Considering MMC process as a simple connection of above adders, if at least one of the intermediate soft adders satisfies the condition for error generation, an error is generated and propagates to the final output of MMC process. Therefore, VOS error (E_i) can be described as a non-positive discrete random variable with supports at multiples of -2^{R_S} which has following characteristics: i) $|E_i| \leq SAD_i$, and ii) $E_i^{vos} = 0$ for $SAD_i < 2^{R_S}$.

2.2.2 The Soft Error Model

Note that for the R_I to be $K+1$ for one addition, the result, S, has to include a 1 followed by K consecutive 0s (from the $(k+1)-th$ bit to the $(k+K)-th$ bit), i.e., $S = m2^{K+k} + U$, where $m > 0$ and $U < 2^k$. Here $m2^{K+k}$ are numbers that are always zero below $K+k-th$ bit and have at least one non-zero bit above. Thus if i) $S = m2^{K+k} + U$ where $K \geq R_S$, and ii) $a_{k+1} = 1$ (automatically $b_{k+1} = 1$), then an error with magnitude 2^{R_S+k} is generated.

Since a_{k+1} corresponds to a lower significance as compared to S, we can assume that S and a_{k+1} are independent. And if we assume i) $P(S = m2^{R_S+k}+U)$ is similar for all U (if A and B are Gaussian or Laplacian whose variance is in the order of 2^{R_S+k}, S has smooth distribution for small range, e.g., 2^k, around $m2^{R_S+k}$), and ii) that $p(a_{k+1} = 1) = \frac{1}{2}$, we can model the error in a soft adder as:

$$P(error = 2^{R_S+k}) = \sum_{m=1}^{T} P(S = m2^{R_S+k})2^{k-1}, \tag{2.1}$$

where $T = \lceil \frac{2^n}{2^{R_S+k}} \rceil$ and n is the width of the adder.

If at least one of the intermediate soft adders satisfies the condition for error generation, an error occurs. This condition depends on R_S and on the outputs of intermediate adders, each of which is the partial SAD corresponding to the $l-th$ node ($PSAD_l$). Thus we can model the error, E_i, given SAD_i, if we know the characteristics of $PSAD_l$. In [34], given a final SAD (SAD_i), the characteristics of $l-th$ partial mean absolute distance ($PMAD_l$) are modeled as random variables with mean $\frac{SAD_i}{M^2}$. Mean absolute distance (MAD) is defined as $\frac{SAD}{M^2}$, where M^2 is the number of pixels. Since $PSAD_l$ is a multiple of $PMAD_l$, a model for $PSAD_l$ can be easily derived for a given final SAD. We can then derive a model for E_i;

this will be the probability of error for the single-adder case summed over all intermediate adders, i.e.,

$$P(E_i = 2^{R_S+k}) = \sum_{m=1}^{T} \sum_{l=1}^{M^2} P_{PSAD_l}(PSAD_l = m2^{R_S+k})2^{k-1} \tag{2.2}$$

Here we assume that an error occurs on only one intermediate node so that $E_i = 2^{R_S+k}, k = 0, 1, ...$ From our simulations, we observed that the probability of errors occuring in more than one node tends to be negligible for the Vdds of interest (e.g., the probability could be less than 10^{-5}). Because our objective using this error model is to model averaged errors in the system level, the effect of small probability of having errors in more than one nodes, could be negligible. Therefore, we make use of this single-error assumption as it simplifies the modeling and the experimental results validate it.

Any given SAD_i value can be produced by many different combinations of intermediate node outputs ($PSAD_l$). Since E_i depends on those intermediate computations, different E_i can be generated for a given SAD_i value. Thus, we model E_i as a random variable which is independent of SAD_i if $SAD_i \geq 2^{R_S}$ (if $SAD_i < 2^{R_S}$ then $E_i = 0$), but cannot take arbitrary values. Note that our simulation shows that correlation between SAD_i and E_i is very small (< 0.03).

2.3 Modeling the effect of soft errors on the coding performance

Using the error model for E_i in Section 2.2.2, we now propose a model for coding performance degradation (e.g., the increased rate for fixed QP, ΔR, which is a well known metric for the video coding performance evaluations), as a function of low level parameters (i.e., R_S and the characteristics of SAD). We first briefly explain how errors in SAD_i lead to the system performance degradation.

For each macroblock of size $M \times M$ in the current frame, the MMC process computes the matching metric for each candidate block in the reference frame's search window. As in Section 1.2, SAD_i and SAD'_i denote the fault-free and faulty SADs, respectively. The residual block's distortion (as measured by the SAD) increases by $E_{SAD} = SAD_{MV_f} - SAD_{min}$. This increase in distortion (E_{SAD}) may lead to rate increases for a given QP, which we propose to model using the quadratic (Q2) model [13]. This model is the representative of currently proposed approaches, has been applied to existing video encoding standards (i.e., ITU-T H.264/MPEG4 AVC [3]), and tends to be accurate for large data sets, such as Group of Pictures (GOPs) or complete sequences (its accuracy increases with the number of frames being modeled). The main Q2 model is defined as follows:

$$R = S_1 \frac{MAD}{QP} + S_2 \frac{MAD}{QP^2}, \tag{2.3}$$

where R is the rate, MAD is the mean absolute difference of the prediction residual ($MAD = \frac{SAD}{M^2}$), QP is the quantization parameter and S_1, S_2 are parameters to be estimated. One can see that the Q2 model can be rewritten as a linear function

of SAD for a fixed QP. Taking the derivative of both terms, the following relation holds for an SAD increase (E_{SAD}) and rate increase (ΔR):

$$\Delta R = X_1 E_{SAD} \tag{2.4}$$

where X_1 is a parameter to be estimated for each set of frames (i.e., one GOP or whole sequence). Therefore if we can model E_{SAD}, we can also model the effect on rate, ΔR. Now we focus on modeling E_{SAD} as a function of R_S and input characteristics, with our proposed model for E_i.

For each SAD_i there is a possibility that due to an error, i will be chosen as the MV ($P(i = MV_f)$), instead of the correct vector. When this happens, the SAD of the prediction residual increases by $SAD_i - SAD_{min}$. But only $SAD_i \geq 2^{R_S}$ can result in a error and thereby lead to an erroneous MV choice. Thus we define Q as the set of SAD_i for which errors can occur so that i is selected as MV_f ($Q = \{i | 2^{R_S} \leq SAD_i\}, N_Q = |Q|$). Then E_{SAD} can be written as follows:

$$E_{SAD} = \sum_{i \in Q} (SAD_i - SAD_{min}) P(i = MV_f), \tag{2.5}$$

To estimate E_{SAD} according to (2.5), we evaluate $P(i = MV_f)$ first. For $i = MV_f$, SAD'_i needs to be smaller than all other SAD'_w, which can be stated as follows:

$$P(i = MV_f) = \prod_{w \neq i} P(SAD'_w > SAD'_i) \tag{2.6}$$

This equation is based on the observation that each error (E_w) is almost independent of corresponding SAD value (SAD_w) (see Section 2.2.1). For i to be

chosen as the MV_f, SAD'_i should be the minimum among all SAD'_w, so that $SAD'_i < SAD_{min}$ and thus the following holds:

$$P(i = MV_f) = \int_0^{SAD_{min}} P(SAD'_i = x) \prod_{w \neq i} P(SAD'_w > x) dx \tag{2.7}$$

Here we can derive a simple expression of $P(SAD'_w < x)$ which is a linear function of x, assuming that $\sum_{l=1}^{M^2} P_{PSAD_l}(PSAD_l = m2^{R_S+k})$ (probability that the value of one of the intermediate outputs ($PSAD_l$) is $m2^{R_S+k}$) is independent of k and takes a constant value p_0:

$$P(SAD'_w < x) \approx \frac{x}{2^{R_S+1}} p_0. \tag{2.8}$$

By using (2.8) in (2.7), we can obtain:

$$P(i = MV_f) \approx \frac{1}{N_Q}(1 - (1 - \gamma \frac{SAD_{min}}{2^{R_S}})^{N_Q}), \gamma = \frac{p_0}{2}. \tag{2.9}$$

Note that p_0 is the probability that one of the intermediate nodes have a certain value, e.g., $m2^{R_S+k}$. Generally p_0 will depend on the MMC architecture. The parallel architecture has more balanced dynamic range than the serial architecture [9]. Compared to the parallel architecture that has M parallel groups of "leaf" adders and M "central" adders where each group of leaf adders consists of $M - 1$ adders and computes the sum of M absolute difference (AD) values, the serial architecture has M^2 serially connected adders for SAD computation. Since only the nodes that reach certain large values can lead to an error, intermediate values needs to be above a certain level. In the serial architecture, successive nodes have increasingly large values, whereas in a highly parallel architecture, those nodes with large

average values can be found only towards the end of the computation tree, e.g., central adders. For example, for a given final SAD, in serial architectures, around half of the intermediate nodes (i.e., $\frac{M^2}{2}$) can have mean values larger than $\frac{SAD}{2}$, while in parallel architectures, this is only possible for central adders, of which there are M. Therefore, it is more likely that one of the intermediate nodes will have an error in the serial architecture, because only nodes whose output can be greater than $m2^{R_S+k}$ can lead to an error. Thus p_0 corresponding to a serial MMC architecture is larger than for a parallel one. These p_0 values can be precalculated for given SAD_w, R_S, and MMC architectures. Now we apply (2.9) to (2.5) to get the following expression for E_{SAD}:

$$E_{SAD} \approx (\overline{SAD_Q} - SAD_{min})(1 - (1 - \gamma \frac{SAD_{min}}{2^{R_S}})^{N_Q}), \tag{2.10}$$

where $\overline{SAD_Q}$ is a mean SAD value over set Q, i.e., the set of SAD_i whose value is larger than 2^{R_S}, and $\Delta R = X_1 E_{SAD}$. In this equation, intuitively E_{SAD} is small when R_S is large and N_Q is small, where N_Q is small for large R_S (less SADs are above 2^{R_S}). Thus for large R_S, E_{SAD} tends to be small. For the special case where $N_Q = 0$, i.e., when all SADs are below 2^{R_S}, E_{SAD} is equal to zero.

2.4 Simulation and Discussion

To evaluate our proposed model, the Foreman and Stefan sequences were tested. We simulated the effect of a series of R_S values using an H.264/AVC baseline profile encoder with full search (FS)/enhanced predictive zonal search (EPZS) ME algorithms and serial/parallel MMC architectures. Only 16×16 block partitions

and a single reference were considered for ME; QP was fixed and rate distortion optimization was turned on. We assign 15 frames to each group of pictures (GOP), and use an IPPP GOP structure. We collect real rate increase (ΔR) data by encoding each GOP with/without errors (for $R_S = 5, 7, ...15$ and Vdd_{crit} corresponding to $R_S = 16$).

Using the analysis in Section 2.3, we then estimate ΔR as a function of R_S for a given sequence before encoding so that we can control Vdd in a optimal fashion during the encoding process, thus saving power. A normal video encoder optimization scheme only considers rate and distortion. But in encoding scenarios that require low power consumption, e.g., hand held devices, we need to take power consumption into consideration by adding this to the cost function. To estimate ΔR, we need information about SAD_i (SAD value for each MV candidate in one macroblock). To get real ΔR data, we would need to encode the same sequence several times with different Vdd settings. Instead, using a E_{SAD} model, we can estimate ΔR in a efficient way, if there is given information about SAD_i (SAD value for each MV candidate in one macroblock).

Since information about SAD_i is not available before encoding, we provide a heuristic method to gather this information. Here we assume that changes in SAD_i within a GOP are negligible. Based on this assumption, we encode the first frame within a GOP with $R_S = 16$ to obtain statistics about SAD_i. With this information, we compute E_{SAD} for all R_S values. Then we encode the first frame with $R_S < 16$ to obtain ΔR, so that we have $X_1 = \frac{\Delta R}{E_{SAD}}$. Fig. 2.3 shows the variations of the real and estimated ΔR as a function of R_S, which will be useful to design a power control mechanism. This result shows that we can precisely estimate ΔR with our analytical model in the R_S range of interest. Note that some

curves in the results are non-monotonic functions of R_S for lower R_S. For lower R_S, errors in motion estimation increase so that the vectors found in the search process tend to have higher SAD value. Thus it is more likely that Intra coding will be used instead of ME-related modes so that ΔR stops increasing and gets into constant ΔR stages where Intra coding is chosen always. During this transition, insufficiently accurate RD decision techniques can lead to non-monotonicity.

With a model for ΔR, we can select Vdd using various optimization techniques, such as those based on Lagrange multipliers [52]. We select a budget constrained method, i.e., select Vdd such that estimated ΔR is less than a threshold for rate change (ΔR_{th}). Now we encode the remaining frames with the selected Vdd above. Using this algorithm, we can dynamically select Vdd for each GOP based on input characteristic with a slight additional complexity, but with potentially large power savings. Fig. 2.4 highlights the potential for the power savings in the ME process using DVS; setting $\Delta R_{th} = 0.1$, leads to 37% power reduction when using EPZS and the parallel MMC architecture. Considering that a significant percentage of power consumption is due to the ME process (e.g., 66%-94% in an MPEG-4 encoder [33]), total power savings within the video encoding system can be significant. Note that the training cost for model estimation, i.e., one extra encoding for one frame out of one GOP (e.g., size of 15 frames) is considered in our simulation results in Fig. 2.4.

By using simulation results and model, we can compare MMC architectures and ME algorithms. The slope of ΔR for serial MMC architecture is larger than that of the parallel architecture, because p_0 of a serial MMC architecture is larger (see Section 2.3). But the saturated ΔR value is similar because it only depends on SAD_{min} and $\overline{SAD_Q}$, which is the same for both cases. Since the EPZS search

strategy uses a good prediction algorithm to select a small number of MV candidates, which are already near the minimum SAD point, EPZS has smaller N_Q and $\overline{SAD_Q}$ than the full search algorithm. Thus ΔR of EPZS is always smaller than that for the full search case. In summary, the EPZS search algorithm is better than full search algorithm and parallel MMC architecture is better than serial MMC architecture. Here, we have not consider the inherent difference in complexity, regularity, and memory usage between EPZS and FS algorithm; FS algorithm has more searching points but has a more regular structure and memory usage than EPZS.

2.5 Conclusion

In this chapter, we applied Dynamic Voltage Scaling (DVS) to Motion Estimation (ME) process. We proposed an analytical model for a rate increase as a function of Vdd and input characteristics, and compared ME algorithms and MMC architectures; EPZS search algorithm and parallel MMC architecture are better than Full search algorithm and serial MMC architecture respectively. Our simulations showed that our model is precise enough to be used to estimate ΔR. Using that model, we also proposed a method to actively control Vdd of the MMC architecture within ME process to save power (37% power saving).

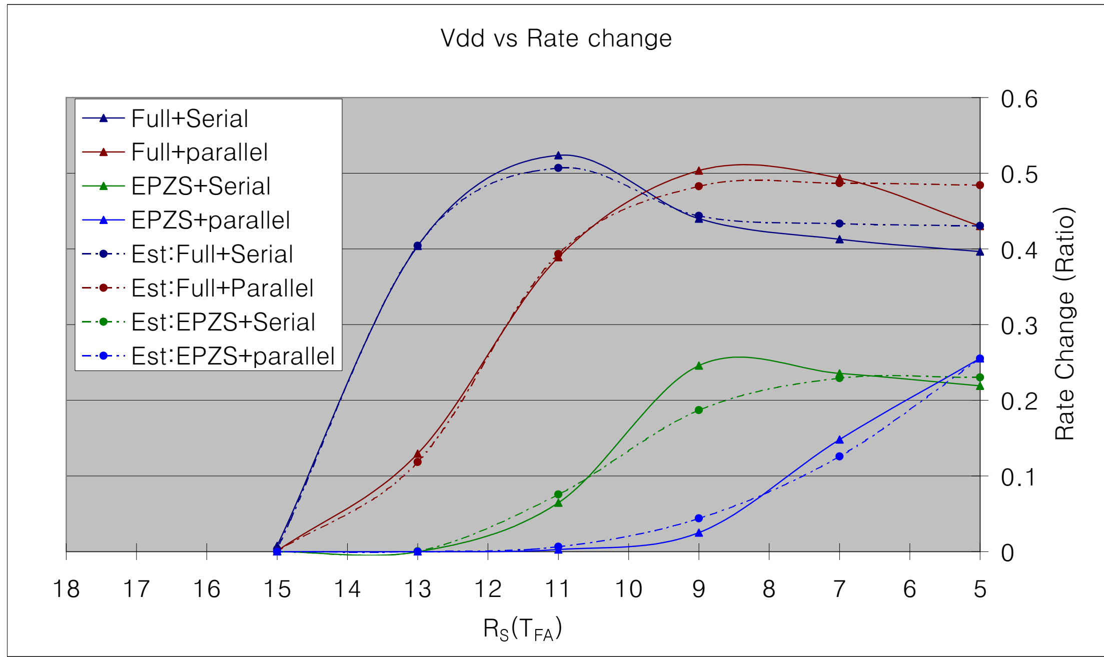


Fig. 2.3: Rate change due to DVS in ratio compared to original rate (dot: estimated data using our model, solid: real data); using FOREMAN sequence, QP=20

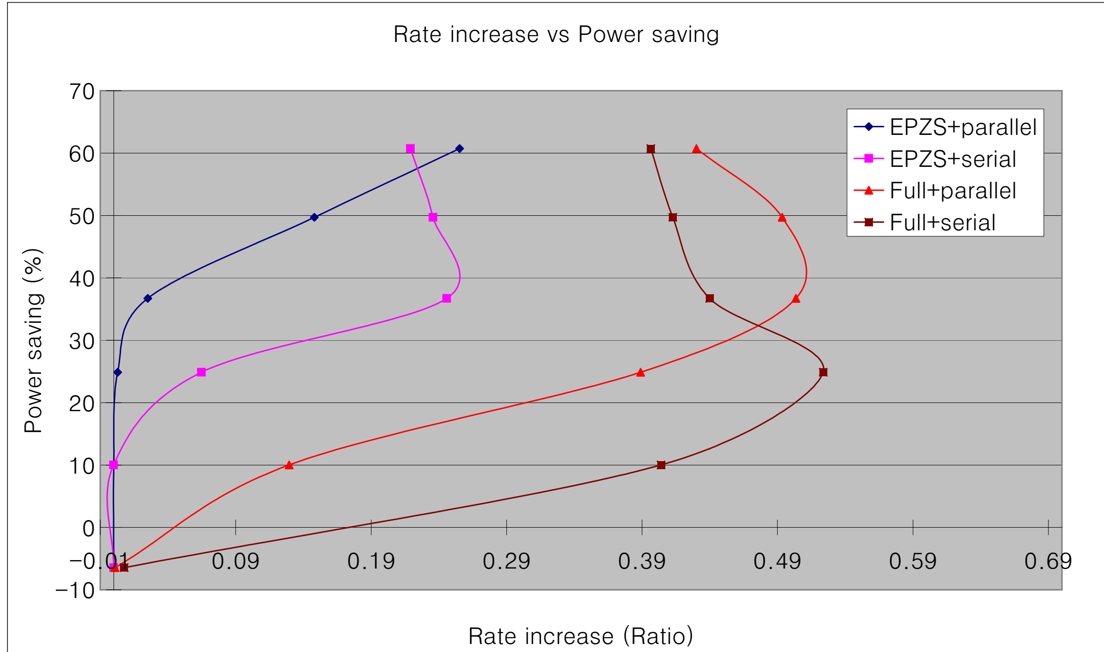


Fig. 2.4: Power saving effect of ME process using DVS for various ME algorithms and MMC architectures; considering redundancy due to encoding one frame of every GOP with Vdd_{crit} and R_S^L for SAD_i information and X_1 estimation, FOREMAN sequence, QP=20, gate parameters ($\alpha = 2.0, Vdd_{crit} = 3.3V, V_t = 0.62V$)

Chapter 3

Low Complexity ME using Multiple Imprecise MMC

3.1 Introduction

In the previous chapter, we introduced the error tolerance (ET) characteristics of motion estimation (ME) and exploited those in order to implement a low complexity ME using one voltage over scaled metric computation. In this chapter, we will discuss more general frameworks for low complexity ME using multiple imprecise metrics, still exploiting the ET characteristics of ME.

3.1.1 Previous work

Algorithmic approaches for power efficient (or low complexity) ME [27, 33, 34, 42] have been studied for a number of years, and these can be classified into roughly two categories; fast search (FS) and fast matching (FM). FS focuses on searching only a small subset of candidates, while FM focus on reducing the complexity of metric computations by using partial metric [34], pixel truncation [27], or new

features [42]. Most of them are based on using a *single* computationally *error-free* metric.

Recently, a new technique that allows an *imperfect* computation within ME, voltage over-scaled (VOS) metric computation [49], was introduced. It has been shown that significant additional power savings are made when the input voltage (Vdd) for the metric computation is set below critical voltage (Vdd_{crit}). A reduction in Vdd by a factor W can lead to power dissipation that is nearly a factor of W^2 lower [28]. Note that above approach was introduced as a feasible solution for *hardware based* variable low complexity computational metric. The lower power consumption that can be achieved comes at the cost of *input-dependent computational errors*; lower input voltage increases circuit delay, and the number of basic operations possible for one clock period decreases, thus generating error. These errors can either i) be concealed by the encoding process (e.g., a motion vector selected to minimize matching metric cost can be correct, even if the matching metric cost itself is incorrect) or ii) lead to "acceptable" degradation (e.g., a small distortion or rate increase) [15].

Other recent work [49, 50] has also proposed two-metric system using VOS as a main computation block and a sub-sampled (SS) version of the original macro-block data as an error concealment computation. Although the idea of using a multiple pairs of identical metrics to reduce complexity was already introduced in [9], using different metrics especially in the context of ME is first proposed in [49], which uses a combination of metrics with algorithmic (e.g., SS) and computational errors (e.g., VOS). Using matching metric cost (e.g. SAD or SSD) computed by these two computation metrics, the "true" matching metric value for each candidate is estimated; a simple heuristic method (e.g., threshold estimator) is proposed to

combine SAD computations provided by the two metrics [49]. In our previous work, as shown in [15], analytical error models for both VOS [15] and SS [34] were used. These models lead to novel techniques for combining the SAD values obtained by the two modules, and preliminary results in [15] outperform the previous threshold method [49].

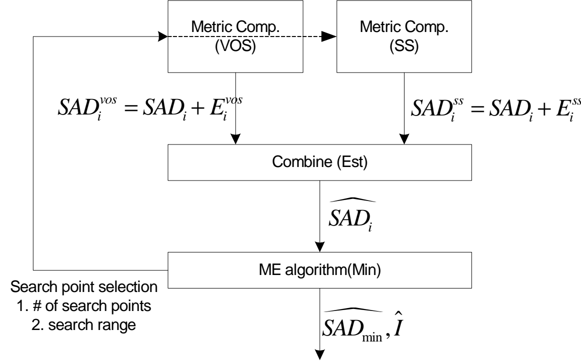


Fig. 3.1: ME with two imprecise metric computations; SAD is used as matching metric cost

Considering these previous works, our goal is to consider, in a more formal and general way, the problem of ME with multiple metric computations. In considering multiple metric systems, there are multiple problems of interest: i) desirable characteristics of multiple metrics, ii) methods to combine those metrics, iii) the frameworks to compare different combinations of metrics and different methods to combine those metrics.

3.1.2 Contribution

Based on these problems, in this chapter, we explore techniques to combine *multiple low complexity* metrics to achieve a low complexity ME. Note that each of those metrics can be computationally with/without errors. The encoding process can

either conceal these errors, or lead to "acceptable" system level degradation [15]. Therefore, we focus on how to trade off complexity and ME system performance degradation, which was not considered in the previous works [15, 16, 49, 50]. Note that this trade off can be fully evaluated using extensive simulations for the possible parameter set.

Within this framework, we first study desirable characteristics of the metrics, i.e., such that the combination of metrics leads to better performance than using each of the metrics independently. In addition, we propose techniques to evaluate the complexity-degradation trade off *without exhaustive simulation*. Furthermore, we propose methods to combine multiple metric computations in order to estimate the best motion vector (MV), which minimizes ME system performance degradation. It is important to note that this approach for estimating the best MV can be used for any combination of metrics as long as the error models are given. As a case study, we applied our framework to combine VOS and SS metrics, whose error models are known (given in [15, 34], for VOS and SS, respectively), to obtain and evaluate MV estimators (e.g., hypothesis testing estimator and MAX). Our results show that our new estimators are better than those previously proposed [49, 50]; our solutions show 78% power savings as compared to 67% in the previous solution, with less than 2% performance degradation. Compared with [49], which considers around 10% performance degradation as an acceptable degradation, we apply a stricter degradation criterion in this work. Therefore, we would expect even greater power savings if 10% performance degradation was considered to be acceptable.

We also show that our proposed guidelines for choosing a good combination of metric computation modules and estimating complexity-degradation trade off,

hold for real scenario. Based on the desirable characteristics of multiple metrics, we introduce a new multiple metric system whose performance is better than the combination of VOS and SS metrics (87% power saving with less than 2% performance degradation). Furthermore, we compare various configurations: ME algorithms (full search, FS, and enhanced predictive zonal search, EPZS [12]), matching metric computation (MMC) architectures (parallel and serial).

Note that some of our techniques may also apply to the environments where multiple different *perfect* metric computations are performed in a noisy environment (e.g., deep submicron noise [45]). With appropriate models, the techniques we discuss may lead to increased resilience with lower overhead, as compared to traditional techniques such as triple modular redundancy [35].

Although our proposed work is designed to be used in hardware-based video encoders, it could also be useful in the context of general purpose processors for which power control is enabled (see [23] for an example). Since ME is performed at the encoder, our work is primarily applicable to scenarios where power-constrained devices (e.g., cellphones) are used for video capture and encoding.

The framework for multiple matching metric ME is introduced in Section 3.2. In Section 3.3, we propose desirable characteristics of metrics, and a method to estimate complexity-degradation trade off. Two general methods to estimate the best MV are introduced in Section 3.4, and we introduce case studies to evaluate our proposed solutions in Section 3.5 and Section 3.6. Simulation results follow in Section 3.7. Finally, in Section 3.8, we compare ME algorithms and MMC architectures.

3.2 Framework for multiple matching metric Motion Estimation

The ME process comprises a search strategy and a matching metric computation (MMC). There are several types of hardware architectures to compute the matching metrics, with different levels of parallelism. Among those, we consider both serial and parallel architectures, where M^2 basic operations (i.e., absolute difference computations and additions) are performed to compute SAD between two $M \times M$ macro-blocks (see Fig. 3.2) [39].

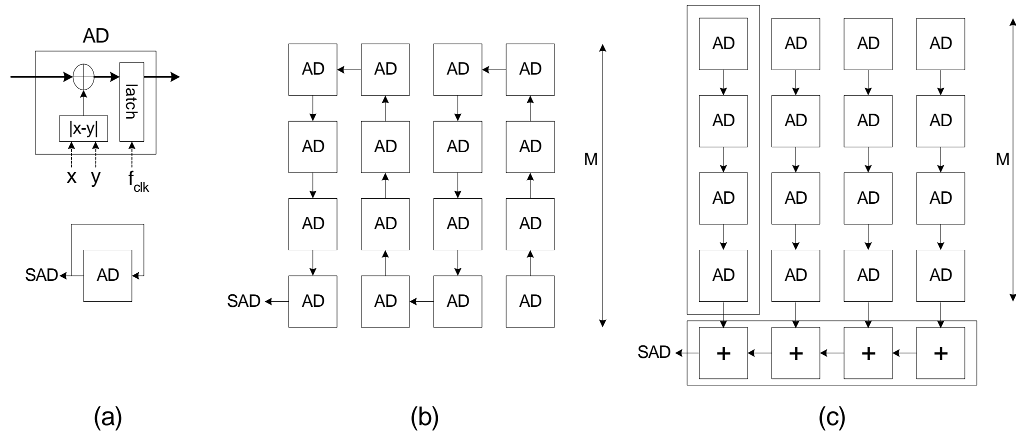


Fig. 3.2: MMC architectures [33, 50]. (a) Structure of AD and serial architecture with one AD, (b) Serial architecture with M^2 ADs, (c) Parallel architecture with M^2 ADs

For each $M \times M$ macro-block in the current frame, we assume that there are N candidates. The SAD for the i-th candidate MV is denoted as SAD_i. Let I be the index of the best MV, in terms of SAD ($I = \arg\min_i(SAD_i)$), so that the minimum SAD is $SAD_{min} = SAD_I$.

For each candidate vector i, assume we use U SAD computations $(SAD_i^1, \ldots, SAD_i^U)$ each corresponding to a low complexity metric. Note that SAD_i can be considered to be an unknown parameter that has to be estimated based on those U SAD results. Assume that the computation of k-th metric can be adjusted based on a selectable parameter π_k which leads to complexity $C_k(\pi_k)$ and computation error $(E_i^k = SAD_i^k - SAD_i)$. A model is assumed to be known for this error, which could be available in a closed form (e.g., Laplacian distribution for SS metric) or as an empirical distribution (e.g., discrete distribution for VOS metric).

This error model can be used to derive the probability that true SAD_i is x_i when $SAD_i^k = x_i^k$ is measured, i.e., $P(SAD_i = x_i | SAD_i^k = x_i^k)$. Assuming the errors of different metrics are uncorrelated, likelihood for SAD_i given all measurements of i-th candidate MV ($L(x_i)$) can be derived as follows:

$$L(x_i) = \prod_{k=1}^{U} \frac{P(SAD_i = x_i | SAD_i^k = x_i^k)}{P(SAD_i = x_i)}. \tag{3.1}$$

Note that our analysis in Section 3.3 will support the fact that uncorrelated errors are desirable. Once the likelihoods for all candidate vectors $(L(x_i), i = 1, \ldots, N)$, are available, they are used to find the best MV, which we denote with index $\hat{I}$ the most likely candidate based on the observations and model. The $L(x_i)$ can be used either i) *jointly* to estimate $\hat{I}$ based on all the likelihoods (see Fig. 3.3 (a)) or ii) *separately* to first estimate each SAD_i ($\widehat{SAD_i}$) based on the $i - th$ likelihoods and then to choose $\hat{I}$ which minimizes $\widehat{SAD_i}$ (see Fig. 3.3 (b)). Using the first approach, we describe an *optimal* method to estimate $\hat{I}$, which minimizes performance degradation. For the second approach, we introduce a good estimator of each SAD_i. We will describe the details of those two approaches in Section 3.4.

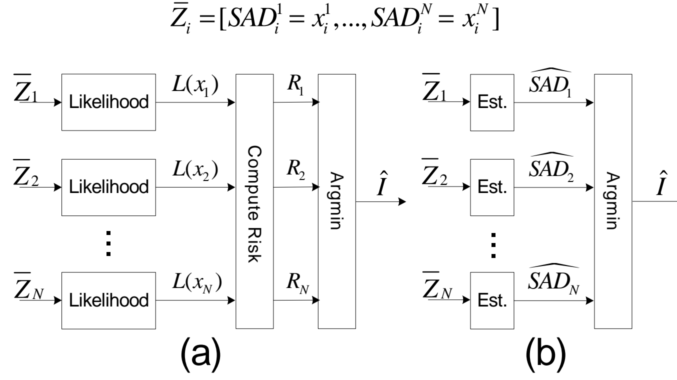


Fig. 3.3: Frameworks for MV estimation: (a) hypothesis testing using likelihoods jointly, (b) estimation of each SAD_i followed by minimum operation

If $I \neq \hat{I}$, the residual block's distortion (as measured in SAD) increases by $E_{SAD} = SAD_{\hat{I}} - SAD_I$. This increase in distortion (E_{SAD}) may lead to ME system performance degradation (e.g., a rate increase (ΔR) for fixed QP). We now consider how to trade off the total complexity of multiple metric computations and the corresponding performance degradation.

For the k-th metric, the corresponding ME system performance degradation is denoted as $D_k(\pi_k)$, which along with $C_k(\pi_k)$ is assumed to be known for all possible values of π_k. For the combination of metrics, the total complexity and degradation pair (C_T, D_T) will be computed. C_T is the sum of the complexity of each metric, $C_T(\pi_1, \ldots, \pi_U) = \sum_k C_k(\pi_k)$, for the complexity of our interest (e.g., power consumption). The total degradation $(D_T(\pi_1, \ldots, \pi_U))$ can be computed using exhaustive simulations (e.g., encoding video source with /without errors). Since this would be impractical, we investigate efficient methods to estimate the total degradation *without exhaustive simulation* in Section 3.3.

With all (C_T, D_T) pairs available, we can construct a complexity-degradation (CD) curve, which is assumed to be convex, based on the experimental data (see Fig. 3.6). The complexity degradation function is defined as follows:

$$C(D) = \min C_T(\pi_1, \ldots, \pi_U), \quad D_T(\pi_1, \ldots, \pi_U) \leq D, \tag{3.2}$$

where the CD region is the closure of the set of achievable complexity degradation pairs (C, D). Based on the CD curve, we compare different combinations of metrics and MV estimation methods. Also we find an optimal operating point using optimization methods, e.g., budget constraint approach; if the maximum performance degradation is D_{th}, the minimum complexity point $C(D_{th})$ can be found. Note that different applications of video coding lead to different threshold values.

Based on the above framework, we propose solutions for three problems of interest. Firstly, we focus on identifying desirable characteristics for the multiple metrics to be combined, i.e., characteristics that can lead to better CD performance. Secondly, we propose methods to estimate the total degradation *without exhaustive simulations*, which leads to efficient CD curve estimation. Finally, we describe how to find the best MV using likelihoods which are derived from known error models of metrics.

3.3 Desirable characteristics of multiple metrics and CD curve estimation

The analysis in this section is based on the following assumptions. We define $\varepsilon_{\mathbf{i}} = [E_i^1, ..., E_i^U]$, the vector of errors in the SAD_i computations. We assume that

$\varepsilon_{\mathbf{i}}$ can be modeled as jointly Gaussian with zero means, variance σ_k^2 for E_i^k, and covariance σ_{kl} between E_i^k and E_i^l. Then each SAD_i^k is Gaussian with mean SAD_i and variance σ_k^2. When we draw sample vector $\mathbf{m_i} = [SAD_i^1 = x_i^1, ..., SAD_i^U = x_i^U]$, the likelihood $L(x_i)$ can be computed as jointly Gaussian with mean vectors $\mathbf{m_i}$ and covariance matrix $\mathbf{K_i}$ whose components are variance σ_k^2 and covariance σ_{kl}.

Based on these likelihoods, we estimate the best MV using the framework in Fig. 3.3(b), where the maximum likelihood (ML) estimator is used to estimate each SAD_i. In this setting, a desirable property for the metrics is associated with a better estimate of each SAD_i, because better $\widehat{SAD_i}$ leads to better MV estimate, and less performance degradation.

The quality of $\widehat{SAD_i}$ is measured in terms of mean squared error ($E(|\widehat{SAD_i} - SAD_i|^2)$). We first find an ML estimate of each SAD_i which can be derived as a linear combination of U SAD computations, $\widehat{SAD_i} = \sum_{k=1}^{U} t_k SAD_i^k$. Note that for the Gaussian case, the ML estimator is the same as the minimum mean squared error (MMSE) estimator, and to satisfy $E[SAD_i] = E[\widehat{SAD_i}]$, the t_ks should satisfy the following: $\sum_{k=1}^{U} t_k = 1$ and $0 \leq t_k \leq 1$ [20]. The quality of that estimate can be derived as follows:

$$E(|\widehat{SAD_i} - SAD_i|^2) = \sum_{k=1}^{U} t_k^2 \sigma_k^2 + \sum_{k}^{U} \sum_{l,l \neq k}^{U} 2 t_k t_l \sigma_{kl} \tag{3.3}$$

Clearly, in (3.3), *less correlation* between errors of metrics (σ_{kl}), results in better SAD_i estimation, and would be a desirable property. Note that the errors produced by the VOS and SS metrics are almost uncorrelated (less than 3% correlation in our

experiments). If metrics are uncorrelated, $\sigma_{kl} = 0$ and the corresponding optimal t_k is derived as $t_k = \frac{\sigma_k^{-2}}{\sum_{r=1}^{U} \sigma_r^{-2}}$, in which case (3.3) can be simplified as

$$E(|\widehat{SAD_i} - SAD_i|^2) = \frac{1}{\sum_{k=1}^{U} \sigma_k^{-2}} \quad (3.4)$$

(3.4) shows that metrics with small error variance dominate the quality of SAD_i estimate. Therefore, metrics with *small error variance* are desirable. Also combining multiple metrics whose computation errors have *similar variance* is desirable. This is because using an additional metric with higher variance increases the complexity without having a significant impact on the quality of estimate of SAD_i.

Using the above analysis with the same assumptions, Algorithm 1 can be used to estimate the CD curve of a specific combination of metrics *without exhaustive simulation.*

Algorithm 1 CD curve estimation method

Step 1. Compute merged complexity: $C_T = \sum_{k=1}^{U} C_k$
Step 2. Compute merged variance: $\sigma_T^2 = \frac{1}{\sum_{k=1}^{U} \sigma_k^{-2}}$.
Step 3. Estimate function $D(\sigma^{-2})$ using known $(D_k(\pi_k), \sigma_k^2)$ pairs of each metric for all possible π_k, assuming $D(\sigma^{-2})$ is a convex curve (our simulations support this).
Step 4. Estimate merged degradation: $D_T = D(\sigma_T^{-2})$.

Up to now, we estimated each SAD_i separately. From now on, we consider all SAD_i estimates *together* in order to minimize degradation. The best MV is the one that minimizes SAD_i, and therefore $\widehat{SAD_j} \geq \widehat{SAD_I}$ for $j \neq I$ ensures a correct MV selection is made. Therefore, larger estimation errors are tolerable for larger SAD_i, and it also means that small estimation errors are desirable for small

SAD_i. Note that the combination of VOS and SS metrics satisfies this condition (this will be discussed in Section 3.5.1).

3.4 MV estimator

In the section, we describe how to find the best MV ($\hat{I}$) using the likelihoods either jointly or separately. Note that each likelihood is computed using known error models, which can be obtained by online/offline model estimation [16]. Also note that this approach works for any given error models.

3.4.1 Hypothesis testing

We first consider a MV estimator which uses joint likelihood to find the $\hat{I}$ such that the performance degradation is minimized (see Fig. 3.3 (a)). In this approach, we assume that distribution of SAD is unknown, and we use a uniform prior for SAD for the simple implementation of the estimator. Based on that, we first derive the a posteriori probability of each SAD_i, i.e., the probability that $SAD_i = x_i$, given vector of computed SADs $\mathbf{\bar{Z}_i} = [SAD_i^1 = x_i^1, ..., SAD_i^U = x_i^U]$, using Bayes theorem, an uniform prior for SAD_i, and the likelihood function $L(x_i)$ in (3.1):

$$P(x_i) = P(SAD_i = x_i|\mathbf{\bar{Z}_i}) = Q \prod_{k=1}^{U} P(SAD_i^k = x_i^k|SAD_i = x_i), \qquad (3.5)$$

where Q is a normalization factor. Based on these a posteriori probabilities of all SAD_is, the optimal way to estimate the MV can be defined in a hypothesis testing

framework. In this framework, we have N hypotheses, where the i-th hypothesis corresponds to choosing the index i as best MV:

$$H_i : \hat{I} = i, i = 1, \ldots, N \tag{3.6}$$

For each H_i, the risk R_i, i.e., the expected value of performance degradation when i is chosen as best MV, is computed using a posteriori probabilities of all SAD_is. We model performance degradation to be proportional to E_{SAD}, where E_{SAD} is the difference between real and estimated SAD_{min}, i.e., $E_{SAD} = SAD_{\hat{I}} - SAD_I$. For each H_i, $SAD_{\hat{I}} = SAD_i$, therefore the MV estimator based on a hypothesis testing can be formulated as follows:

$$\hat{I} = \arg\min_i R_i, \quad R_i = E[SAD_i - SAD_I | \bar{\mathbf{Z}}_\mathbf{1}, ..., \bar{\mathbf{Z}}_\mathbf{N}], \tag{3.7}$$

where expectation is over SADs given vectors of computed SADs, $\bar{\mathbf{Z}}_\mathbf{1}, ..., \bar{\mathbf{Z}}_\mathbf{N}$. This R_i can be defined as,

$$R_i = \int \ldots \int_{-\infty}^{\infty} (x_i - min(x_1, \ldots, x_N)) \prod_{r=1}^{N} P(x_r) dx_1 \ldots dx_N. \tag{3.8}$$

To implement a MV estimator with hypothesis testing framework, our focus is on how to derive a posteriori probability of each SAD_i ($P(x_i)$) based on SAD computations. We will provide an example of implementation based on the combination of VOS and SS metrics, for which error models have been proposed, in Section 3.5.

3.4.2 Sub-optimal MV estimator

The MV estimator based on the hypothesis testing is optimal in terms of minimizing performance degradation; however, it is complex. We now propose an alternative sub-optimal MV estimator, shown in Fig. 3.3 (b). This approach starts by estimating SAD_i which will be denoted as $\widehat{SAD_i}$. Based on these estimates, the candidate with minimum $\widehat{SAD_i}$ is chosen, i.e.,

$$\hat{I} = \arg\min_i \widehat{SAD_i}. \tag{3.9}$$

For the estimator of each SAD_i, a threshold estimator [49, 50] was previously proposed as a solution, and ML estimator was used in Section 3.3. In those estimators, the focus is on minimizing estimation errors of each SAD_i, and those are parametric solutions. As an alternative, in the following Section 3.5, we propose the MAX estimator, which is non-parametric, and therefore more robust to inaccurate model estimation [16].

3.5 Case study: MV estimators using VOS and SS metrics

In this section, as a case study, we apply CD curve based framework to the case where VOS and SS metrics are used (refer to Fig. 3.1). We first briefly describe error models for VOS [15] and SS [34] metrics. Based on those, we introduce how to implement a MV estimator based on hypothesis testing, and then we propose a desirable estimator of each SAD_i (e.g., MAX estimator) for a sub-optimal MV

estimator in Fig. 3.3 (b). Finally, we evaluate those MV estimators, compared to the previously proposed threshold estimator.

3.5.1 Characteristics of VOS and SS errors

We denote SADs corresponding to the i-th candidate, which are computed using the SS and VOS metrics (see Fig. 3.1), by $SAD_i^{vos} = SAD_i + E_i^{vos}$ and $SAD_i^{ss} = SAD_i + E_i^{ss}$ respectively. The SS error (E_i^{ss}) can be modelled as Laplacian with parameter λ for given SAD_i, where λ is a function of the sub-sampling parameter m; larger m's correspond to larger λ parameters [34].

As discussed in Section 2.2.1, the VOS error (E_i^{vos}) is a non-positive discrete random variable whose values can only be multiples of -2^{R_S}. It has following characteristics: i) $|E_i^{vos}| \leq SAD_i$, and ii) $E_i^{vos} = 0$ for $SAD_i < 2^{R_S}$. We adopt a simplified model to describe VOS errors; denoting $p_0 = P(E^{vos} = 0)$ we assume that the probabilities of nonzero errors are all equal to $\frac{1-p_0}{L}$, where $L = \lfloor \frac{SAD}{2^{R_S}} \rfloor$. Note that typically p_0 is very large for the Vdds in which we are interested, and $p_0 = 1$ for $L = 0$. In short, this simplified model is completely characterized by a single parameter p_0. Experimental results show that it provides a good approximation.

Furthermore, we make the following assumptions. First, as mentioned earlier, SS and VOS errors are assumed to be *uncorrelated* (in our simulation, we observe correlation < 0.03). Second, we can characterize VOS errors as having large magnitude but low rate, compared to SS errors which are relatively small with high rate. Third, parameters (λ and p_0) have significant dependency on SAD (e.g.,

small λ and large p_0 are observed for small SAD) with, in some cases, an order of magnitude difference for different SAD values.

3.5.2 Threshold estimator

In the previous work [49, 50], a threshold estimator of each SAD_i is proposed, which is defined as follows, given a choice of threshold, Th:

$$\widehat{SAD_i} = \begin{cases} SAD_i^{ss}, & if \quad SAD_i^{vos} - SAD_i^{ss} > Th \\ SAD_i^{vos}, & if \quad SAD_i^{vos} - SAD_i^{ss} \leq Th, \end{cases} \tag{3.10}$$

where Th is defined as:

$$Th = \max |SAD^{ss} - SAD|. \tag{3.11}$$

In this approach the MV which minimizes $\widehat{SAD_i}$ is chosen. Note that the threshold is defined in terms of original SAD, which is not known beforehand, so that an approximate procedure (or training) may have to be developed to estimate it; in [49,50] assuming E^{ss} is a Gaussian random variable, a distribution and threshold were estimated using the training data set. This threshold estimator is a method based on the observation that VOS errors have large magnitude compared to SS errors.

3.5.3 MAX estimator

As an alternative, we propose the MAX estimator, which first estimates SAD_i based on SAD_i^{ss} and SAD_i^{vos} using the following non-parametric equation:

$$\widehat{SAD_i} = SAD_i^{max} = \max(SAD_i^{ss}, SAD_i^{vos}), \tag{3.12}$$

and then finds MV which minimizes $\widehat{SAD_i}$. Like threshold estimator, this uses the same framework in Fig. 3.3(b). Compared with the threshold estimator, MAX estimator is a non-parametric estimator, which is very simple and robust.

To motivate our non-parametric estimator, it can be shown that a positive biased estimator of each SAD_i is desirable to minimize E_{SAD} if that estimator ensures $\widehat{SAD_1} \approx SAD_1$ by exploiting the characteristic of combination of metrics (see Appendix). Our proposed MAX estimator satisfies this condition when the combination of VOS and SS metrics are used. Firstly, it clearly is a positively biased estimator, as the largest computation value is always chosen as an estimate. Secondly, it ensures $\widehat{SAD_1} \approx SAD_1$ because of the following reasons. SAD_1^{max} can be either SAD_1^{vos} or SAD_1^{ss}. Due to VOS errors' non-positive large magnitude, the first case where $SAD_1^{max} = SAD_1^{vos}$, i.e., $SAD_1^{vos} \geq SAD_1^{ss}$, holds if $SAD_1^{vos} = SAD_1$ and $SAD_1^{ss} \leq SAD_1$. Therefore, in this case, SAD_1^{max} is the same as original SAD_1. On the other hands, if $SAD_1^{max} = SAD_1^{ss}$, i.e., $SAD_1^{vos} < SAD_1^{ss}$, the error between SAD_1^{ss} and SAD_1 is very small, because SS errors are very small for the small SAD. Therefore the MAX estimate of SAD_1 is very close to real SAD_1, and a positive biased estimator such as MAX is shown to be desirable.

3.5.4 MV estimator using hypothesis testing

We now propose a MV estimator based on the hypothesis testing for the combination of VOS and SS metrics. For this estimator, a posteriori probabilities of all SAD_is ($P(x_i)$) are derived based on error models for SS and VOS metrics, which can be described using p_0 and λ. To estimate those parameters, online adaptation is used, which does not require extra data for model estimation. Firstly, to derive $p_0 = P(SAD^{vos} = SAD)$, we exploit the following observations. VOS errors are *non-positive* and tend to be large compared to SS errors, therefore $SAD^{vos} \geq SAD^{ss}$ holds, if $SAD^{vos} = SAD$ and $SAD^{ss} \leq SAD$, where $P(SAD^{ss} \leq SAD) = 0.5$. Based on this approximation, p_0 can be derived as follows.

$$p_0 = 2P(SAD^{vos} \geq SAD^{ss}) \tag{3.13}$$

With the same reasoning, we can derive λ as follows:

$$\lambda = E[SAD^{vos} - SAD^{ss} | SAD^{vos} \geq SAD^{ss}] \tag{3.14}$$

These parameters can be updated online, because real SAD values are not needed in those estimations. Using above parameters, we can derive $P(x_i)$ as follows:

$$P(x_i) = \begin{cases} P_{vos}(x_i - SAD_i^{vos}), & if \quad SAD_i^{vos} \geq SAD_i^{ss} \\ P_{ss}(x_i - SAD_i^{ss}) P_{vos}(x_i - SAD_i^{vos}), & otherwise, \end{cases} \tag{3.15}$$

where $P_{ss}(x)$ is Laplacian with a parameter λ and $P_{vos}(x)$ is a discrete random variable with $P_{vos}(0) = p_0$ and $P_{vos}(k2^{R_S}) = \frac{1-p_0}{L}$. Originally these error models depend on SAD. However, using SAD dependent models would be too complex,

thus we provide a practical alternative to ignore parameters' SAD dependency, whose performance is shown to be very close to the ideal one (i.e., using SAD dependent error models) in our simulation. We first introduce "pre-screening" with a MAX estimator, whose performance is quite good. When we select the smallest P candidates ($P << N$) based on MAX estimates (SAD_i^{max}), it is highly probable that the best MV can be found among those P candidates. Now that we only consider the P smallest candidates, the range of SADs of interest is very small compared to original one, thus, it is reasonable to assume that, within this range, parameters (λ, p_0) do not change. Note that reducing the number of candidates to P reduces the complexity of risk computation dramatically. Considering above, our implementation uses Algorithm 2.

Algorithm 2 Algorithm for a MV estimator with hypothesis testing framework

Step 1. Choose P smallest candidates with pre-screening with MAX estimator.
Step 2. Update parameters (e.g., p_0 and λ) using SAD computations from those candidates.
Step 3. Compute R_i in (3.8), for those candidates.
Step 4. $\hat{I} = \arg\min_i R_i$

3.6 Case study: Desirable characteristics of multiple metrics and CD curve estimation

In this section, we first introduce a new combination of metrics, namely two SSVOS metrics, based on the design guidelines proposed in Section 3.3. As can be seen in Fig. 3.4, each SSVOS metric computes SAD using each sub-sampled set of pixels with a fixed sub-sampling parameter $m = 2$ [34], where each computation is voltage over scaled with a input voltage parameter Vdd [15]. This new combination of

metrics satisfies the guidelines proposed in Section 3.3. First, SS metric has small error variance for given complexity, compared to VOS metric. Second, the two metrics have similar error variances, by applying the same Vdd and m on both SSVOS metrics. Third, metrics based on VOS and SS have small estimation errors for small SAD_i. Note that if we directly combined two SS metrics with m=2 and without VOS, then we ensure an operating point without performance degradation.

We now show that our proposed design guidelines in Section 3.3 hold for real scenarios where SS and VOS are combined (see Fig. 3.1) and where two SSVOS are combined (see Fig. 3.4), even though simple assumptions (e.g., Gaussian errors and ML estimate of SAD) were used to derive those guidelines.

We first verify that the performance of a combination of metrics is dominated by metrics with low variance of errors (refer to Fig. 3.6). For the right two CD points of VOS CD curve (with complexity values 0.2 and 0.29, respectively), VOS errors have much smaller variance than SS errors (e.g., $\frac{\sigma_{vos}^2}{\sigma_{ss}^2} < 0.04$), and therefore the performance is dominated by VOS metric, i.e., merged degradation (D_T) of the combination of VOS and SS metrics is close to that of VOS metric. The opposite happens when SS errors have smaller variance than VOS errors (e.g., $\frac{\sigma_{vos}^2}{\sigma_{ss}^2} > 4$ for the leftmost CD point of VOS CD curve).

We then show that metrics with similar error variances are desirable by studying the combination of two SSVOS metrics (see Fig. 3.4). In this combination, we can set separate parameters for different SSVOS metrics (e.g., $Vdd1$ and $Vdd2$ for the first and second SSVOS metrics, respectively). But using different parameters does not improve performance, as compared to using the same parameters for both metrics (e.g., the same Vdd for both SSVOS metrics), i.e., CD curve using different $Vdd1$ and $Vdd2$ for each metric is not better than one that using the same Vdd.

Furthermore, we show that the quality of proposed CD curve estimation based on the above described simplifying assumptions is sufficiently good, for both combinations of metrics. As can be seen in Fig. 3.5, estimated CD curve based on above assumptions is very close to real one, therefore, we can use this estimated curve to evaluate specific combination of metrics and to decide on the operating point. From this, we can also conclude that our proposed design guidelines in Section 3.3 are sufficiently effective at least for the metrics we have, because these guidelines are based on the same analysis which is used for the CD curve estimation.

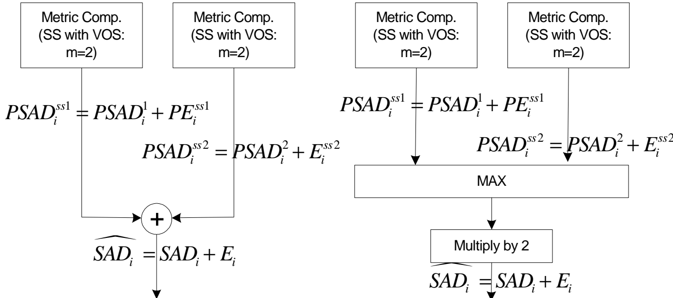


Fig. 3.4: Two SSVOS metrics combined with adder and MAX

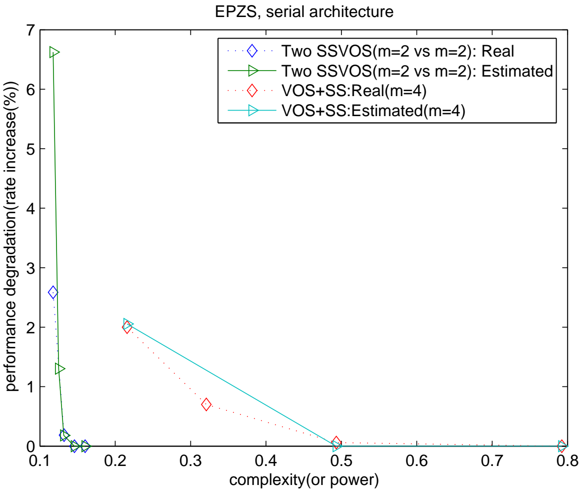


Fig. 3.5: Estimation of CD curves

3.7 Simulation Results

In this section, we first evaluate the performance of our proposed estimators: i) the MV estimator using hypothesis testing, and ii) MAX estimator, and compare their performance to that of threshold estimator. For our experiments we use the FOREMAN sequence and an H.264/AVC baseline profile encoder with full-pel EPZS ME algorithms and serial MMC architectures. Only 16×16 block partitions and a single reference were considered for ME, and rate distortion optimization (RDO) was turned on. We assign 15 frames to each group of pictures (GOP), and use an IPPP GOP structure. We collect normalized rate increase (using Bonjegaard metric [6]), by encoding each GOP with/without errors for different Vdd ($R_S = 9, 11, 13, 15$ where $R_S = 16$ for error free operation), $QP = 17, 22, 27, 32$, and $m = 4$. Note that we also tested for $m = 2, 8$ but the performance of $m = 4$ is the best in terms of complexity-degradation trade off. We also estimate complexity based on power consumption for each operating point. We estimate P_{orig}, the power consumptions by the normal (i.e., not voltage over scaled) metric which operates on Vdd_{crit}, as [49, 50]:

$$P_{orig} = (1 + G)C_{vos}Vdd_{crit}^2\alpha_1 f_{clk}, \tag{3.16}$$

where C_{vos} and α are gate parameters and f_{clk} is a operating frequency of adders in the MMC architecture. Similarly, P_{vos} and P_{vos+ss}, the power consumption with a VOS metric and a combination of VOS and SS metrics, respectively, can be estimated as:

$$P_{vos} = (1 + G)C_{vos}Vdd_{vos}^2\alpha_1 f_{clk}, \tag{3.17}$$

$$P_{vos+ss} = (1+G)C_{vos}Vdd_{vos}^2\alpha_1 f_{clk} + (1+G)C_{ss}Vdd_{ss}^2\alpha_2\frac{f_{clk}}{g(m)}, \qquad (3.18)$$

where $g(m)$ is equal to m and $\frac{2m}{m+1}$ for the serial and parallel MMC architectures respectively. Note that first and second terms of P_{vos+ss} are power consumption from VOS and SS metrics respectively, and we normalized P_{vos+ss} and P_{vos} dividing it by P_{orig}.

With these settings, we compare the performance, in terms of CD characteristics, of our proposed estimators to the threshold estimator, when VOS and SS (denoted here, VOS+SS) metrics are used. Obviously, the performance of a hypothesis estimator, the optimal one, is the best among three estimators. MAX estimator's performance is almost the same as that of the hypothesis estimator and better than threshold estimator, especially in low complexity mode. In higher complexity mode, the performance of MAX estimator is very close to the other estimators. When we assume 2% rate increase as an acceptable degradation threshold (note that this is much lower than that in [50]), we can save 78% power by using either the hypothesis testing or the MAX estimator with serial MMC architecture, as compared to 67% savings for the threshold estimator (see Fig. 3.6). Note that the performance of the combination of VOS+SS metrics is better than that achieved when using the VOS metric by itself, i.e., 52% power saving for one VOS metric.

Also the performance of two SSVOS modules (denoted 2SSVOS here) is better than the VOS+SS approach (87% power saving with less than 2% degradation when MAX estimator used as a combination method; see Fig. 3.6). Note that this combination can be extended to more than two SSVOS metrics, but it is fairer

to compare 2SSVOS to VOS+SS, because the area complexity of VOS+SS is the same as that of 2SSVOS, if the implementation method in [49, 50] is used.

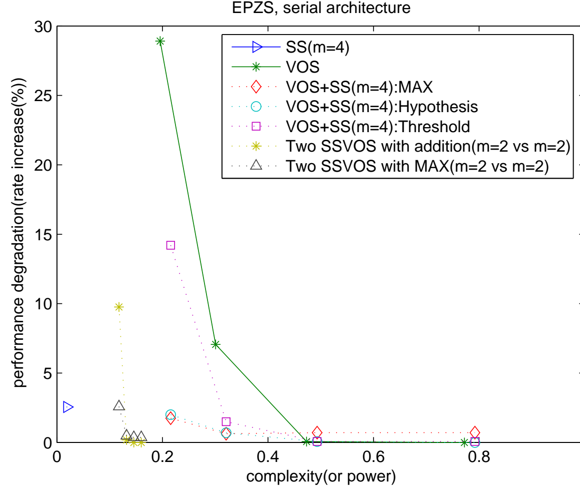


Fig. 3.6: Comparison of MV estimators and metric combinations, FOREMAN sequence, gate parameters ($\alpha = 1.2, Vdd_{crit} = 3.3V, V_t = 0.62V$)

3.8 Performance under different ME algorithms and MMC architectures

In this section, we compare the performance when different ME algorithms (e.g., EPZS and FS) are used. EPZS uses a good prediction algorithm to select a small number of MV candidates, with most of these candidates having relatively good matching accuracy. Thus EPZS shows more resilience to soft errors due to imprecise computations than the FS approach, i.e., a lower Vdd can be used for EPZS, resulting in better power efficiency. For example, when we use a VOS metric with EPZS the power savings is 52%, while for FS the corresponding savings is 23%,

with 2% rate increase as an acceptable degradation threshold (see Fig. 3.7). Note that EPZS is also preferred for the other combinations for metrics.

Varatkar and Shanbhag [49, 50] used a three step search (TSS) algorithm [31], which is shown to be less error tolerant than EPZS [10], so that overall rate increases would be worse if TSS were used instead of EPZS. Note that we do not consider the inherent difference in regularity, and memory usage between algorithms which is not easy to quantify; FS algorithm has more searching points but has more regular structure and memory usage than EPZS.

As for the choice of MMC architecture, it is not fair to compare the complexity-degradation trade-off of parallel and serial architectures directly, because those are different in area complexity and operating throughput. For an $M \times M$ block size, a parallel approach requires M^2 basic modules (e.g., AD which consists of an absolute difference computation and an addition; see Fig. 3.2), while a serial approach can be implemented using only one basic module [49, 50]. Therefore, the area complexity of the serial approach can be smaller than the parallel one.

In case of operating throughput, parallel approach is better. The parallel architecture, as described in Fig. 3.2 (c), requires $2M$ successive additions (i.e., M additions in leaf adders and M additions in central adders), while serial one requires M^2. Thus for the same sampling time (T_S) for each adder, operating throughput of a parallel architecture is higher than serial one.

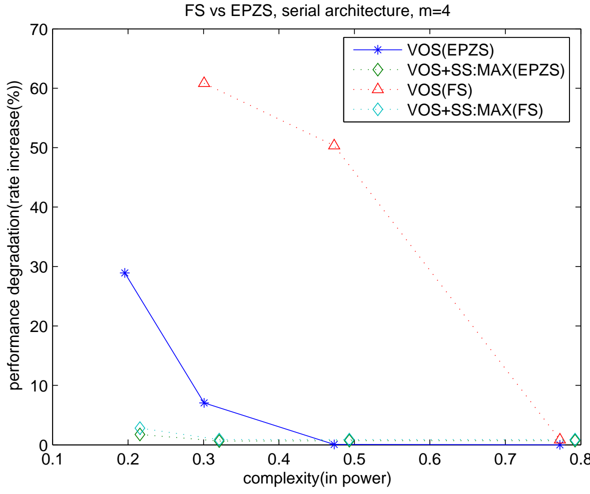
With these factors in mind, we compare the CD characteristic of each MMC architecture. It depends on which combination of metric we choose. When we consider VOS metric, a parallel architecture shows better performance than a serial one (e.g., the parallel architecture results in 78% power savings, while in the serial case the saving is 52% with 2% rate increase as a threshold; see Fig. 3.7). This is

because the parallel architecture has more balanced dynamic range than the serial one. Within MMC architectures, only nodes that reach certain large values can lead to an error, thus intermediate values needs to be above a certain level. It is clear that in the serial architecture successive nodes have increasingly large values, whereas in a highly parallel architecture those nodes with large average values can be only towards the end of the computation tree, e.g., central adders.

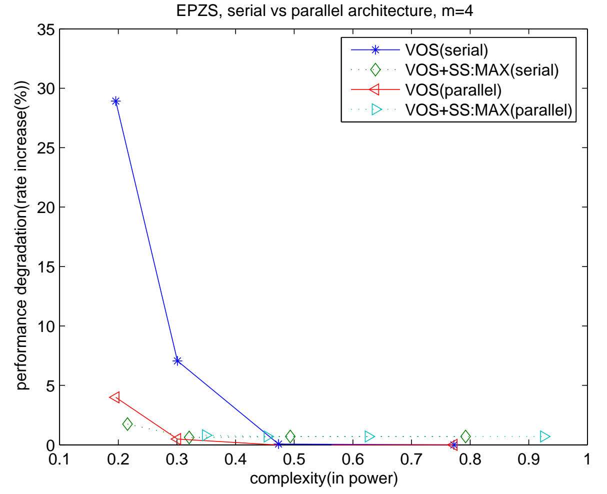
For the combination of VOS and SS metrics, the CD characteristic of the serial architecture is shown to be competitive compared to parallel one although its area is much smaller than that of the parallel one. In serial architecture, SS metric consumes very small portion of power compared to VOS metric (e.g., less than 10%). This is possible because larger T_S can be used for the SS metric, e.g., mT_S for SS metric with sub-sampling factor m, which leads to power savings in a factor of $g(m)$ in (3.18). But for parallel MMC architecture, $g(m)$ is small compared to the serial one, thus the portion of power consumptions of SS metric, compared to VOS metric, is larger, e.g., more than 25%.

3.9 Conclusion

In this chapter, we provided a general framework for multiple matching metric motion estimation (ME), which considers the trade off between complexity and ME system performance degradation. Within this framework, we discussed (i) desirable characteristics for these metrics, (ii) how to estimate complexity-degradation characteristics, and (iii) how to combine multiple metric computations to estimate best motion vector (MV). Furthermore, we compared the performance of different ME algorithms and matching metric computation architectures. As a case study,



(a) ME algorithms



(b) MMC architectures

Fig. 3.7: Comparison ME algorithms and MMC architectures with $m = 4$, FOREMAN sequence, gate parameters ($\alpha = 1.2, Vdd_{crit} = 3.3V, V_t = 0.62V$)

we applied above framework to the combination of voltage over scaled (VOS) and sub-sampled (SS) metrics. We also introduced a new two metric system (i.e., two SSVOS metrics) whose performance is better than the combination of VOS and SS metrics (87% power saving with less than 2% degradation).

Chapter 4

Error Tolerant DCT

4.1 Introduction

In the previous chapters, we studied the error tolerance within motion estimation, and exploited the property for reducing power consumption of ME module. In this chapter, we focus on the error tolerance of a very common component of multimedia compression systems, namely the discrete cosine transform (DCT). This transform is used in video coders, e.g., MPEG [37], and image coders, e.g., JPEG [36], and similar linear block transforms are used in most recent compression systems, such as ITU-T H.264 [3]. In all these systems the transform is followed by *quantization*, thus, while we consider faults in the transform operation, our analysis considers the impact of the faults *after quantization*.

To provide some intuition about the impact of faults in the DCT, consider Table 4.1, where we show the average peak signal to noise ratio (PSNR) degradation produced by single stuck-at faults (SSFs) at the input of a DCT block, i.e., inserting SSFs into the bus input lines of a DCT module. PSNR is a well accepted

objective quality metric used in image/video compression. While quality differences as high as $0.5dB$ are often difficult to perceive in typical image and video coding applications, we will see later in this chapter that the errors introduced due to faults are different from typical errors introduced by lossy compression and can be visible even when degradations below $0.5dB$ occur. Thus we will need to take into account both maximum error and probability of error measurements in order to determine whether faults are acceptable.

We can see that some faults generate acceptable degradation. For example, if we set the acceptable PSNR degradation to be less than $0.2dB$, we see that more than half the faults at the input are acceptable. Observe also from Table 4.1 that a fault's impact depends on the choice of quantization parameters, so that, as expected, a given fault in the transform computation will tend to have a lesser impact as the quantization becomes coarser (i.e., at lower bit rates). Intuitively, both hardware faults and quantization contribute to additional distortion in the decoded images, but as quantization distortion increases the additional distortion due to the fault becomes negligible. This will be further discussed in Section 4.2.1.

Our goal is to define a methodology to discriminate between acceptable and unacceptable faults in systems comprised of a DCT followed by quantization. To do so, we propose tools to (i) estimate the effect of individual faults at the system output, and (ii) decide on metrics and thresholds of acceptable degradation, which will be used to determine whether each specific fault is acceptable or not. This will require analyzing specific system architectures and also taking into account the characteristics of the input (i.e., the statistics of typical image data), as well as specific image/video quality requirements in the application. Note that this framework can be easily extended to other compression systems that make use

Bit/R	1	4	8
0	-0.0030	-0.0009	0.0008
1	-0.0056	-0.0007	0.0002
2	-0.0154	-0.0022	-0.0009
3	-0.0489	-0.0083	-0.0008
4	-0.1943	-0.0257	-0.0082
5	*-0.6406*	-0.1066	-0.0293
6	*-2.6987*	*-0.4066*	-0.1308
7	*-5.3202*	*-1.8567*	*-0.3713*

Tab. 4.1: Changes in decoded image quality due to a SSF at the input of DCT. The quality change is measured for a test image by computing the difference in the PSNR of the decoded image obtained from a fault-free JPEG encoder and that produced by a faulty JPEG encoder. A negative value corresponds quality degradation. Bit 0 corresponds to the least significant bit of an input pixel. The parameter R controls the quantization level, with larger R corresponding to coarser quantization (i.e., lower bit-rate and worse decoded image quality.)

of linear transforms followed by quantization, e.g., JPEG2000 system which uses a wavelet transform. With this methodology we then perform fault analysis to categorize possible faults into two sets, i.e., the set of acceptable and unacceptable faults, respectively. Then we seek to generate testing methods to classify systems with unknown faults into acceptable and unacceptable systems. As compared to the previous works on testing for ET [30, 43], one novelty in this work is that we consider typical input statistics, instead of assuming a simple uniform distribution of inputs [30, 43]. Moreover our work proposes an analytical method to estimate sub-module level error rate and error significance characteristics, which is required for generating test vectors. Our results are specific to the analysis of linear transforms followed by quantization, and our acceptable degradation thresholds take into account the perceptual effect of faults on typical image and video compression applications.

Our results show that a significant fraction (e.g., over 50% in some cases) of interconnection faults within the DCT system are acceptable in a JPEG encoder (operating at any of its admissible compression rates). Our acceptability criteria considers block based distortion, as well as the rate at which block based visible artifacts occur. Our testing method has quite good testing coverage (e.g., over 98% in most cases), and due to the high portion of acceptable faults, we may obtain a significant increase in yield rate (e.g., as much as double the original yield-rate in some cases).

This chapter is organized as follows. In Section 4.2, we propose a general framework for analyzing DCT followed by quantization and provide tools to quantify the effect of faults at the output. In Section 4.3, we propose new metrics and thresholds which consider system analysis and human visual perception. Based on those metrics, we also perform fault analysis, with results shown in Section 4.4. Finally, in Section 4.5, we propose systematic testing methods, provide a comparison of testing qualities and costs, and estimates of the resulting yield rate increase.

4.2 System level error tolerance in DCT and quantization

The DCT is one of the most popular invertible linear transforms that are used to extract meaningful frequency information from signals. Note that in this work, we focus on 8×8 DCT which is used in most of the DCT based video/image codec, e.g., JPEG and MPEG. This can be easily extended to cases with other transform sizes. Fig. 4.1 shows the basic structure of a 2D separable DCT system [40], which

can be implemented using two 1D DCT systems and some memory. The 1D DCT is composed of parallel processing element (PE) modules, where each PE calculates the dot product between 1D input vector and one of the 1D DCT basis vectors $\bar{\mathbf{C}}_{\mathbf{i}}$, which will be denoted DC (lowest frequency) and $AC1$ to $AC7$ (with $AC7$ being the highest frequency basis vector). These separable and dot product based architectures are also used in other linear transforms (i.e., wavelet transforms), and therefore we can easily extend our work to those transforms.

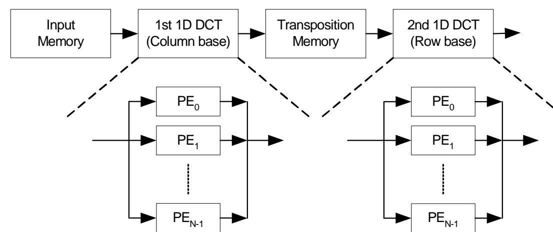


Fig. 4.1: DCT block diagram

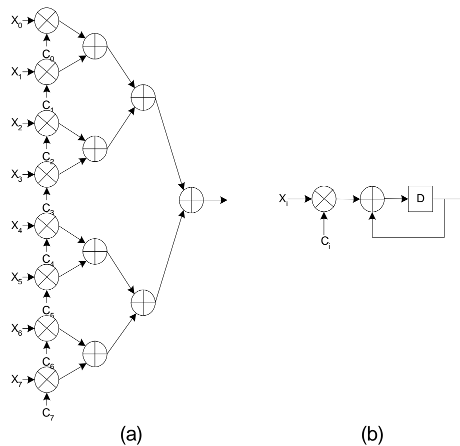


Fig. 4.2: PE architectures; (a) non-sequential, (b) sequential

Among the various structures proposed for PE_i [40], we select two representative ones which have different degree of parallelism; one with a sequential multiplication and accumulation (MAC) architecture and the other with non-sequential

full parallel architecture (see Fig. 4.2). These structures are popular for their simplicity and regularity. In addition, other structures for PE are very similar to ours in the sense that they also use tree based structures, and thus, our work can be easily extended to those structures.

We make the following assumptions. First, our work will be focused on the interconnect faults that affect the data transfer within and between PEs, therefore, we will assume that the carry propagations within multiplication and addition operations are error-free. These error-free processes can be achieved by well-known self checking design techniques [26,46]. Second, we will assume that the faults in the interconnects are single-stuck-at 0 (SSA0) or single-stuck-at 1 (SSA1) [29] faults[1], which cause the given data line to produce a constant value (0/1) independent of other signal values in the circuit. Note that the SSA fault model covers 80-90% of the possible manufacturing defects in CMOS circuits [47].

4.2.1 DCT and Quantization

The input to the system (see Fig. 4.3) is a vector $\bar{\mathbf{X}}$, which we assume drawn from a vector distribution that can be statistically characterized, e.g., by its covariance structure. We can define the set of possible faults, or fault space, $\mathbf{F}$, by analyzing the architecture of the system. Assume there is a single fault $f_i \in \mathbf{F}$ in the transform and denote its faulty (vector) output $\bar{\mathbf{Y}}'$. $\bar{\mathbf{Y}}$ denotes the output of the fault-free system when the input is $\bar{\mathbf{X}}$.

[1] We focus our discussion on these faults but the quality assessment tools we propose could also be applied in other scenarios, e.g., when distortion is added by soft errors

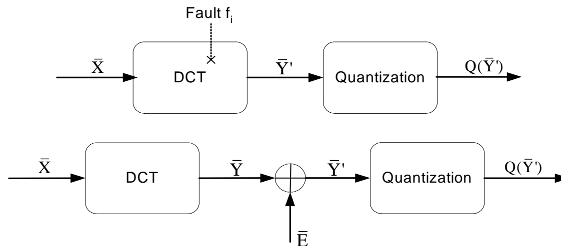


Fig. 4.3: DCT and Quantization

To analyze the effect of fault f_i, we first simplify the problem by viewing its effect as an error term $\bar{\mathbf{E}}$ added to the fault-free output $\bar{\mathbf{Y}}$. Clearly, $\bar{\mathbf{E}}$ is a deterministic function of f_i, $\bar{\mathbf{X}}$, and the structure of the DCT. Since we consider invertible transforms, there is a 1-to-1 mapping between $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$, and thus, $\bar{\mathbf{E}}$ is not independent of $\bar{\mathbf{Y}}$.

In lossy compression applications, this transform is followed by quantization. As scalar quantization is normally used, each component of $\bar{\mathbf{Y}}$ (or $\bar{\mathbf{Y}}'$) is independently quantized. $Y(u, v)$, $E(u, v)$, and $Y'(u, v)$ denote the (u, v)-th component of the vectors $\bar{\mathbf{Y}}$, $\bar{\mathbf{E}}$, and $\bar{\mathbf{Y}}'$, respectively, with $u, v = 1 \ldots N$, with N the vector dimension. When considering individual components, it is reasonable to assume that $E(u, v)$ will be independent of $Y(u, v)$, even though $\bar{\mathbf{Y}}$ and $\bar{\mathbf{E}}$ are dependent. It is because, for a specific value of $Y(u, v)$, there are many possible values of $E(u, v)$, which depend on the $Y(k, l)$, $(u, v) \neq (k, l)$. In what follows, we make the assumption that $E(u, v)$ is a random additive error term independent of $Y(u, v)$. We have verified that this is a reasonable assumption for typical systems (in our simulation, correlation between E and Y is less than 4% when typical images are applied to the DCT system). For convenience, in what follows we focus on one component and drop the frequency index (u, v) if needed.

In SSA0 faults, a specific bus line (say the p-th) is modeled to be always set to 0, so that whenever that line was meant to have a value of 1, an error with magnitude 2^p is added to or subtracted from the original value. The same holds for SSA1 case except that error occurs when the bus line is set to 0. Clearly, if one of these faults affects the computation of a frequency term in the DCT, two similar blocks may suffer very different errors (i.e., no error vs. 2^p error) in a case that they differ in the specific faulty bus line. We also can see that SSA0/1 faults within specific PE or bus lines do not affect *all* output components (e.g., faults within PE_i of 1-st 1D DCT generate errors in $Y(i, v), v = 1, \ldots, N$). Note that these errors are also affected by the quantization operation.

4.2.2 Quantization Analysis

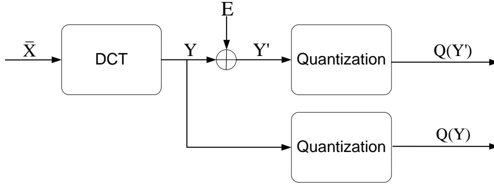


Fig. 4.4: Quantization Analysis

Our focus now turns to analyzing how an error introduced by a fault (E) leads to error *after quantization* for each scalar output Y. Let E and Y be discrete and continuous random variables, respectively, with known pmf/pdf. Let the quantization step size be Δ, and define $D_C^2 = |Q(Y') - Y|$, $D_C^1 = |Q(Y) - Y|$, and $\Delta D = D_C^2 - D_C^1$. D_C^2, D_C^1, and ΔD represent distortion due to both the error

E and quantization, due only to quantization, and due only to E after quantization, respectively, for each output component Y. We will validate the use of those metrics and the use of $L1$ distance in Section 4.3.1.

To facilitate the analysis, we will represent E as follows:

$$E = L\Delta + E',$$

where $L = \lfloor \frac{E}{\Delta} \rfloor$ is a integer and $0 \leq E' < \Delta$.

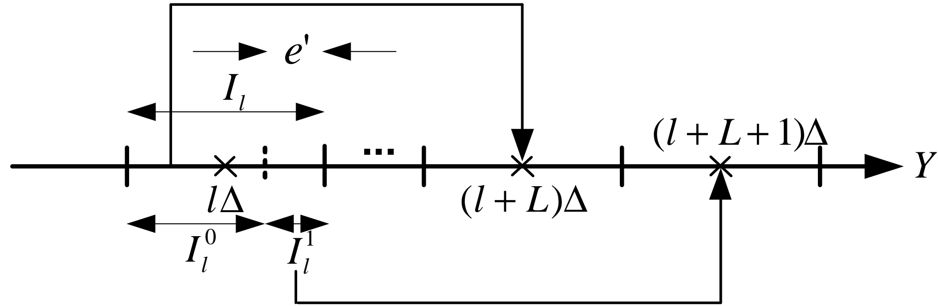


Fig. 4.5: Quantization Analysis. Y is quantized into bins of size Δ. If we focus on one bin (I_l), $Q(Y) = l\Delta$ is the center of that bin. $Y' = Y + e$ (where $e = L\Delta + e'$) now belongs to a different quantization bin. There are two cases for a given error e. If $Y \in I_l^1$ then $Q(Y') = (l + L + 1)\Delta$. Alternatively, $Y \in I_l^0$ leads to $Q(Y') = (l + L)\Delta$.

Fig. 4.5 illustrates the relationship between $Q(Y')$ and $Q(Y)$ for a particular value of the error e. This error can also be written as $e = L \cdot \Delta + e'$. As can be seen in the figure, for $Q(Y) = l\Delta$, and for a given error, $Q(Y')$ can take two different values, depending on the original value of Y. More formally (refer to Fig. 4.5):

$$Q(Y') = \begin{cases} (l + L + 1)\Delta, & if \quad Y \in I_l^1 \\ (l + L)\Delta, & if \quad Y \in I_l^0, \end{cases} \tag{4.1}$$

where $I_l = [l\Delta - \frac{\Delta}{2}, l\Delta + \frac{\Delta}{2}]$, $I_l^0(e') = [l\Delta - \frac{\Delta}{2}, l\Delta + \frac{\Delta}{2} - e']$, and $I_l^1(e') = I_l - I_l^0(e')$. We make the following observations. Even though $Y' \neq Y$, we will have that both Y and Y' fall in the same quantization bin, so that $Q(Y') = Q(Y)$. In these cases the error is masked by the quantization operation, and thus errors do not occur in all image blocks. Another consequence of this observation is that, as quantization becomes coarser, fewer blocks are likely to exhibit fault-related errors after quantization.

Also we observe that $D_C^2 \geq D_C^1$, i.e., the image produced by the faulty system can only have larger distortion than that produced by the fault-free one, because either $Q(Y') = Q(Y)$ or $Q(Y')$ goes to another bin (in which case distortion is increased).

4.2.3 Analysis of additional error due to fault: ΔD

Using above observations and (4.1), we evaluate the error added at the output due to the fault, i.e., ΔD. Fig. 4.6 depicts the relationship between ΔD and Y for a particular value of the error e.

Considering the situation where $L = 0$, i.e., $e < \Delta$ and $e' = e$, we can observe two different cases as follows (see Fig. 4.6 (a)). If $Y \in I_l^0$, $Q(Y)$ and $Q(Y')$ are in the same bin, so that $\Delta D = 0$. Instead, if $Y \in I_l^1$, then $Q(Y) = l\Delta$ and $Q(Y') = (l+1)\Delta$, so that $\Delta D = 2((l+0.5)\Delta - Y)$. In this case, ΔD at the boundary of I_l^1 is $\Delta D|_{Y=l\Delta+\frac{\Delta}{2}-e'} = 2e'$. Also, as Y increases, ΔD decreases. Similar situations arise for every bin of Y. As can be seen in Fig. 4.6 (b), when L is non-zero, ΔD is just shifted by $L\Delta$, but ΔD is not constant for $Y \in [l\Delta, (l+0.5)\Delta - e']$ ($\Delta D = L\Delta + 2l\Delta - 2Y$). Note that for $Y = (l+0.5)\Delta - e'$ (i.e., boundary of I_l^0

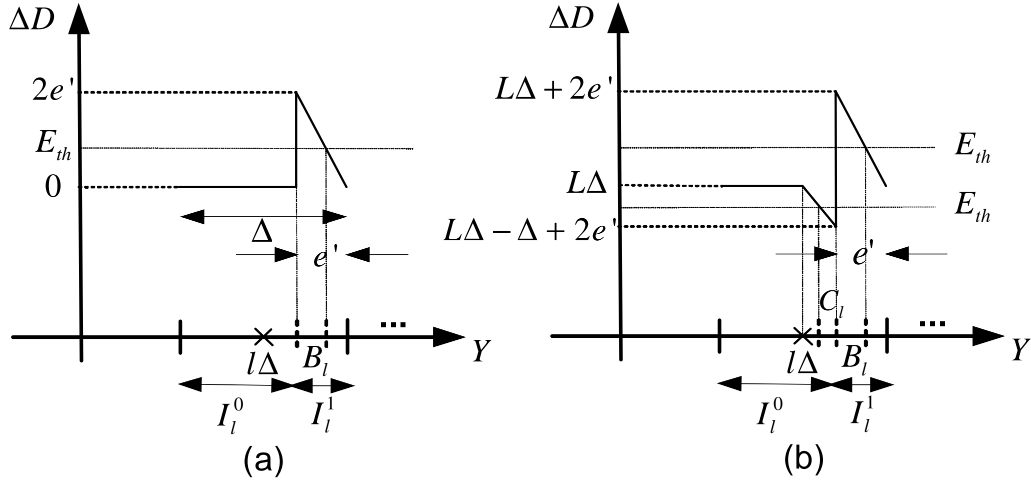


Fig. 4.6: Relation between ΔD and Y for (a): $L = 0$ and (b): $L \neq 0$. At the edge of I_l^1 ΔD has maximum. $\mathbf{B_l}$ is the range of Y in each quantization bin (I_l) where ΔD is larger than error threshold (E_{th}), and $\mathbf{C_l}$ is the range of Y in each quantization bin (I_l) where ΔD is smaller than E_{th}. We need to integrate $f_Y(y)$ over this range to get error rate.

and I_l^1), ΔD abruptly changes in the magnitude of Δ, because $Q(Y')$ jumps from $(l+L)\Delta$ to $(l+L+1)\Delta$ at that Y.

From Fig. 4.6, we derive $P_o = P(\Delta D \geq E_{th})$. To determine P_o, we first select an error threshold for acceptable degradation, which we denote E_{th}. Then we define P_o as follows:

$$P_o = \sum_e P(\Delta D \geq E_{th} | E = e) P_E(e). \tag{4.2}$$

$P(\Delta D \geq E_{th} | E = e)$ can have different values depending on the relative values of error (e) and error threshold (E_{th}). If $e < \Delta$, i.e., $L = 0$ shown in Fig. 4.6 (a), the following holds:

$$P(\Delta D \geq E_{th}|E=e) = \begin{cases} \int_{\bigcup \mathbf{B_i}} f_Y(y)dy, 0 < E_{th} < 2e' \\ 0, E_{th} \geq 2e', \end{cases} \tag{4.3}$$

where $\mathbf{B_i}$ is the range of Y in each quantization bin (Bin_i) where $\Delta D \geq E_{th}$, and is defined as $\mathbf{B_i} = \{Y| \ \ i\Delta + \frac{\Delta}{2} - e' \leq Y < i\Delta + \frac{\Delta}{2} - e' + \frac{2e'-E_{th}}{2}\}$.

If $e \geq \Delta$, i.e., $L \neq 0$ shown in Fig. 4.6 (b), the following holds:

$$P(\Delta D \geq E_{th}|E=e) = \begin{cases} \int_{\bigcup \mathbf{B_i}} f_Y(y)dy, L\Delta < E_{th} < L\Delta + 2e' \\ 0, E_{th} \geq L\Delta + 2e' \\ 1 - \int_{\bigcup \mathbf{C_i}} f_Y(y)dy, L\Delta - \Delta + 2e' < E_{th} \leq L\Delta \\ 1, E_{th} \leq L\Delta - \Delta + 2e', \end{cases} \tag{4.4}$$

where $\mathbf{B_i} = \{Y| \ \ i\Delta + \frac{\Delta}{2} - e' \leq Y < i\Delta + \frac{\Delta}{2} - e' + \frac{L\Delta+2e'-E_{th}}{2}\}$ and $\mathbf{C_i} = \{Y| \ \ i\Delta + \frac{\Delta}{2} - e' - \frac{L\Delta-\Delta+2e'-E_{th}}{2} \leq Y < i\Delta + \frac{\Delta}{2} - e'\}$.

In (4.3) and (4.4), the most computationlly challenging part is integrating over $\bigcup \mathbf{B_i}$ and $\bigcup \mathbf{C_i}$, thus we provide a method to simplify those. Note that $\int_{\bigcup \mathbf{B_i}} f_Y(y)dy \simeq |B_i|/\Delta$ when the quantization bin size (Δ) is much smaller than the dynamic range of the signal, as characterized by σ_{max}, and $\int_{\bigcup \mathbf{B_i}} f_Y(y)dy$ decreases when Δ becomes larger relative to σ_{max}. Thus, we define K as follows:

$$K = \frac{\int_{\bigcup \mathbf{B_i}} f_Y(y)dy}{|B_i|/\Delta}, |B_i| : \frac{e + e' - E_{th}}{2} (interval \ \ of \ \ B_i) \tag{4.5}$$

where, K is a scaling factor which depends on the relation between the dynamic range of the signal (σ_{max}) and the quantization bin size (Δ), and can be written as follows:

- Case 1: $K \approx 1, \Delta << \sigma_{max}$
- Case 2: $K < 1, \Delta = G\sigma_{max}, (G \approx 1)$
- Case 3: $K \approx 0, \Delta >> \sigma_{max}$

And the similar thing can be done for $\int_{\bigcup \mathbf{C_i}} f_Y(y)dy$.

(4.3) and (4.4) allowed us to calculate P_o for one frequency component. Now we need to combine these individual values ($P_o(u, v)$) into a global P_o; this can be done using the summation law:

$$P_o = 1 - \prod_{u=1}^{N} \prod_{v=1}^{N} (1 - P_o(u, v)). \tag{4.6}$$

If we assume $P_o(u, v)$ is small for all u, v, then $P_o \approx \sum_{u=1}^{N} \sum_{v=1}^{N} P_o(u, v)$. Therefore, for a fault to be acceptable with a given block-wise error rate threshold R_{th}, the following will have to hold:

$$\sum_{u=1}^{N} \sum_{v=1}^{N} P_o(u, v) \leq R_{th}. \tag{4.7}$$

Analysis in this section, helps us to understand how quantization affects error tolerance within DCT, in terms of the additional distortion, i.e., ΔD, and error rate, i.e., P_0. Note that this analysis also can be used to estimate ΔD and P_0 without quantized outputs, i.e., $Q(Y)$ and $Q(Y')$. Our preliminary simulations show that results using this anlaysis is close to that of an exhaustive method

using $Q(Y)$ and $Q(Y')$, e.g., around 95% accuracy. But in our fault analysis in Section 4.4, we adopt the exhaustive method for better accuracy.

4.3 Metric and threshold

In this section, we provide new metrics and thresholds to determine which errors are acceptable by evaluating the resulting image/video quality and determining whether the degradation in perceptual quality is "acceptable". Mean squared error (MSE) is a widely used objective quality metric for multimedia compression. While its limitations are well-known [25], it is frequently used to provide a rough comparison between different coding techniques or to drive efficient rate control algorithms. However, in systems suffering from computation errors, MSE is *not* a suitable performance measure.

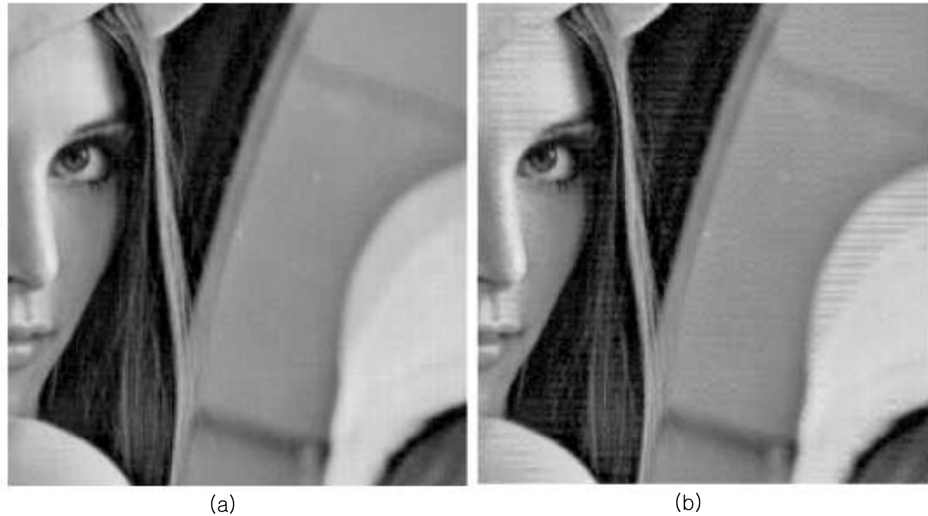


Fig. 4.7: (a): Higher MSE but Acceptable, (b): Lower MSE but unacceptable

To get some intuition about why this may be true in some cases, refer to Fig. 4.7. Fig. 4.7(a) is obtained with a fault-free JPEG encoder, which has higher MSE ($PSNR = 37.9db$). This image, however, is perceptually better than that in Fig. 4.7(b), which has lower MSE ($PSNR = 38.9db$), and was obtained by simulating the presence of faults in the JPEG encoder. In this second image, we can observe a regular error pattern in various areas. In a typical compression scenario, quantization noise tends to affect all components (i.e., *all blocks, all frequencies within a block*). Instead, we are now considering hardware errors that can lead to unevenly distributed artifacts, because they only manifest themselves for certain inputs. We have observed that this is indeed true for hardware errors in the DCT in the context of JPEG coding, i.e., only *certain blocks* and *certain frequency components* within those blocks can be affected, as seen in Fig. 4.7. Clearly, MSE would not be a suitable metric, since a large error in only a few blocks (or frequencies) could be clearly visible by the end user, while still leading to a low overall MSE.

Thus, we provide new metrics which consider the characteristics of a faulty DCT operation, e.g., unevenly distributed artifacts over i) frequency components, ii) blocks or iii) frames (in the case of video). We propose to quantify error significance on a block-by-block basis and then quantify block-wise error rate.

The block-wise error significance (BES) is a block-based objective distortion metric that quantifies the difference between the two decoded images. The block-wise error rate (BER) is the probability that the BES exceeds a certain threshold such that artifacts are visible. Acceptability can then be defined (for a particular application) in terms of the percentage of blocks (in an image or a video sequence) for which visible differences between the images can be tolerated.

Note that here we are considering the additional degradation introduced by hardware faults on decoded outputs. Therefore we will evaluate the error between pairs of decoded images or frames, produced by faulty and fault-free systems. This is different from the standard quality evaluation in the context of compression, where often a decoded image is compared with the original uncompressed one. Our problem is, indeed, more similar to an image ranking problem (where two different decoded versions of an image are compared) [41]. However, unlike typical ranking scenarios, in our case, a single encoder (with fixed coding parameters) is considered, and the image generated by the faulty system cannot be better in quality than that generated by the fault free system.

4.3.1 Metric Selection for Block-wise Error Significance

In this section, we introduce a basic metric for block-wise error significance. Since our ultimate goal is to introduce quality acceptability criteria in automated hardware testing, we consider only objective metrics that take into account the human visual system (HVS) [24]. The faults we consider occur within the DCT block, and thus produce errors in individual frequency components. For this reason, we take Watson's [5,51] techniques as a starting point to define distortion metric for block-wise error significance (BES). In [5], visibility thresholds for each DCT frequency component ($Th(u, v)$) are proposed. If errors in a (u, v)-th frequency component are larger than $Th(u, v)$, regular pattern artifacts which corresponds to (u, v)-th DCT basis can be visible. Using these thresholds, a perceptual objective metric is introduced in [5, 51], which evaluates distance between two images: reference and test image. The DCT is performed on non-overlapping 8×8 blocks in both images

and the absolute differences ($L1$ distance) between corresponding coefficients in the two images are computed. Each resulted difference or error component (e.g., $D_C(u, v)$ for frequency (u, v)) is weighted by the corresponding perceptual threshold ($Th(u, v)$). Then, pooling is performed over the frequency components, first in one block, and then over all blocks in the image pair, to obtain a single distortion metric value. In both cases, Minkowski pooling is used. In our case, we consider block-wise distortion which can be written using only frequency pooling as

$$D = (\sum_{u,v} D_w(u, v)^b)^{\frac{1}{b}}, \tag{4.8}$$

where $D_w(u, v) = \frac{D_C(u,v)}{Th(u,v)}$ is the weighted absolute difference for frequency (u, v), and b is a parameter that controls how much emphasis is given to the largest values. In some cases, $Th(u, v)$ can be modified to incorporate luminance compensation (LC) and contrast masking (CM) [5, 51].

For our purposes, we modify Watson's approach as follows. First, since our goal is to incorporate these thresholds into a testing strategy *we do not use either LC or CM, which would modify the thresholds*, in an image dependent manner. Second, we have noted that faults could lead to large errors in some blocks and no errors in others. For this reason, *we do not perform spatial pooling* of the metric, but measure block by block error. Finally, because faults in the DCT operation can lead to errors being added to only certain frequencies in a given block we use Minkowsky pooling setting $b = \infty$, i.e., the *metric for a block is the largest weighted absolute error in the block, defined as:*

$$BES = D = \max_{u,v} \frac{D_C(u, v)}{Th(u, v)} \tag{4.9}$$

4.3.2 Block-wise error significance and error rate

So far we have discussed basic block-wise distortion metrics that are appropriate for our problem and can be applied to pairs of images. However, unlike traditional image quality assessment problems, we now have three images to consider, i.e., original, fault-free decoded, and faulty decoded images. Thus, we need to choose $D_C(u,v)$ by taking this into consideration.

One possible metric is $D_C^0(u,v) = D_C^2(u,v) - D_C^1(u,v)$, which is already introduced as ΔD in Section 4.2.2. Note that $D_C^0(u,v) \geq 0$, since $D_C^2(u,v) \geq D_C^1(u,v)$. We can think of $D_C^0(u,v)$ as *perceptual distance between* $Q(Y'(u,v))$ *and* $Q(Y(u,v))$ when using the original image component $Y(u,v)$ as a common reference.

Another candidate metric is $D_C^3(u,v)$, which directly evaluates the distance between $Q(Y'(u,v))$ and $Q(Y(u,v))$. This alternative metric can be problematic because of the following reasons. When ranking is considered, we would expect that when $D_C^2(u,v) = D_C^1(u,v) + \delta$ (with $\delta > 0$ small) for specific frequency (u,v), the two decoded images will have almost the same perceptual quality. However, it is possible for $D_C^3(u,v)$ to be large relative to δ when the quantization parameter is large. To see why, consider an example where a single frequency $Y(u,v)$ in a block is affected by a *small* error $E(u,v)$, and $Y(u,v)$ is near the boundary of a quantization bin. Assume the error is sufficiently large so that $Y'(u,v)$ is now in the immediately neighboring quantization bin. Then $D_C^1(u,v) \simeq D_C^2(u,v) \simeq \Delta/2$, so that $D_C^0 \simeq 0$, while $D_C^3(u,v) = \Delta$, which can be large. Based on this, we choose $D_C^0(u,v)$ as our distortion metric between two decoded frequency components. Alternatively, we could use $D_C^3(u,v)$ when Δ is small; however for the simplicity,

we use $D_C^0(u,v)$ for all quantization settings. Using this metric, we can define BES as follows.

$$BES = \max_{u,v} \frac{|Q(Y'(u,v)) - Y(u,v)| - |Q(Y(u,v)) - Y(u,v)|}{Th(u,v)}, \tag{4.10}$$

where $Y'(u,v) = Y(u,v) + E(u,v)$.

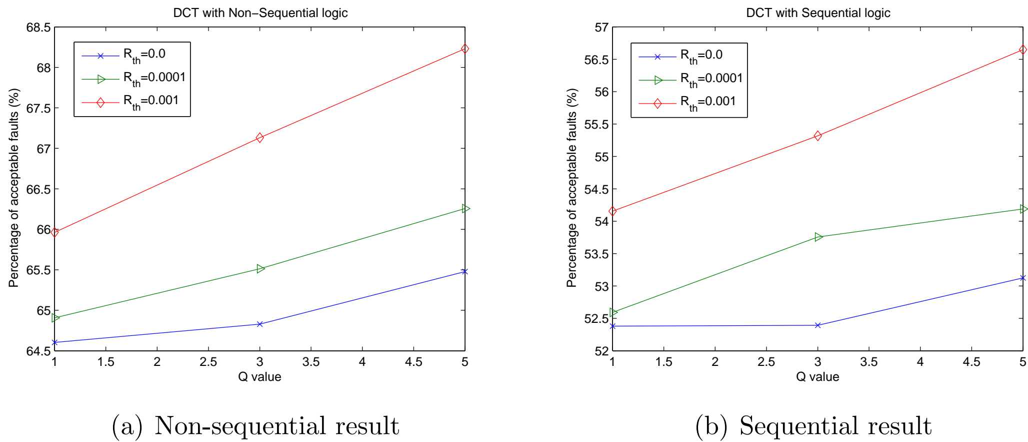
We now define an acceptability threshold for BES. In [41], it was shown that if Watson's distance between two images (in their case, original and decoded) is less than or equal to 1, there are no visible artifacts. This was based on setting the Minkowski frequency pooling parameter b to ∞ (as we do, albeit for different reasons). Equivalently, this condition states that the original and decoded blocks cannot be visually differentiated as long as $D_C(u,v) \leq Th(u,v)$ for all u, v. It is reasonable to use the same condition for our problem, because BES is attempting to quantify perceptual distance between fault-free and faulty decoded image blocks. Simple preliminary results validate this assumption although more thorough perceptual experiments would be needed for a more complete validation. This study falls outside the scope of this thesis. In summary, we define acceptability for an individual block as follows: a faulty decoded image block is acceptable if $D_C^0(u,v) \leq Th(u,v)$ for all (u,v).

We also define an error rate threshold, R_{th}, which is application-dependent (e.g., depending on the resolution of images). Table 4.2 provides examples of threshold values that would guarantee that there is no more than one visible block artifact per two or three frames, at different resolutions. Using above metric and thresholds, we perform fault analysis in the next section.

	CIF(352×288)	QCIF(176×144)	HD(1920×1080)
$N_{8\times8}$	1584	396	32500
$\frac{1}{N_{8\times8}}$	6.3×10^{-4}	0.0025	3.1×10^{-5}
R_{th}	0.0001	0.001	0.00001

Tab. 4.2: Possible error rate threshold (R_{th}) values depending on the resolution values, where $N_{8\times8}$ is the number of 8×8 blocks within one image or frame

4.4 Fault analysis



(a) Non-sequential result

(b) Sequential result

Fig. 4.8: Fault analysis results

In this section, we introduce a method to perform fault analysis for given DCT and PE architecture, quantization parameter set $Q = \{q_i, i = 1, \ldots, N_Q\}$, input set, and threshold values (i.e., R_{th}). The target of this fault analysis is to divide all possible faults ($F = \{f_i, i = 1, \ldots, N_F\}$) into two sets, i.e., the set of acceptable faults F_A and the set of unacceptable faults, F_{NA}, based on acceptability criteria introduced in Section 4.2.

In our work, we are considering a specific input pdf, which captures typical image characteristics (e.g., $Y(u, v)$ can be modeled as independent Gaussian with a mean $m(u, v)$ and a variance $\sigma(u, v)^2$, which is trained using real images). Then

$X(u, v)$ can be expressed as a linear combination of all $Y(u, v)$s, because DCT is an linear invertible transform. As we assumed previously, we simulate the behavior of faults (e.g., SSA1/0 for interconnects within DCT) in a standard JPEG encoder. In addition, we consider $Q = \{1, 3, 5\}$ which corresponds to high, medium, and low quality of decoded images respectively, and $R_{th} = \{0.001, 0.0001, 0.00001\}$ which are chosen based on the resolutions in Table 4.2.

We first gather parameters of image characteristics (i.e., mean and variance matrices for $Y(u, v)$s) by using 10 representative images (i.e., Lenna, Stefan, Boats, Elaine, Couple, F16, Pentr, Pentl, Moon, Chem, and House2). Then, we perform Monte Carlo simulations using this image model. We first generate random input blocks (i.e., $X(u, v)$s) from that model. For each random block (and for each fault), we determine whether error significance (using $Q(Y), Q(Y'), Y$) exceeds the threshold, and if so count this block towards the probability of error. Note that the number of blocks has to be large enough to provide reliable estimates of error rate as compared to R_{th}. For example, if $R_{th} = 0.01$, 1000 inputs are introduced, and a system with a specific fault is declared to be unacceptable if more than 10 error events happen. In this way, the acceptability of all the faults is determined, depending on quantization parameter Q and threshold R_{th}. Our results show that the non-sequential architecture PE is more error-tolerant than sequential one; over 52% of faults are acceptable for a sequential PE architecture, while over 64% of faults are acceptable for a non-sequential one (See Fig. 4.8). Considering the dependency on quantization parameters, more faults are acceptable for coarse quantization parameter, because coarse quantization conceal more errors generated in the faulty DCT system. As for R_{th}, clearly a higher error threshold value leads to more faults being considered “acceptable”. For example, when we change above

parameters (Q, R_{th}) from $(1, 0.0)$ to $(5, 0.001)$, our results show that percentage of acceptable faults increases from 52% to 56% for sequential PE, and from 64% to 68% for non-sequential PE. In this section, we performed a fault analysis on the DCT structures of our interests. The results of fault anlaysis of these structures, i.e., acceptibility of each fault, will be used for test vector generations in Section 4.5.

4.5 Systematic Test Vector generation for DCT

In this section, we introduce a systematic DCT test vector generation method. Traditional testing methods focus on determining the existence of faults in the system by applying testing vectors $T = \{t_i, i = 1, \ldots, N_T\}$, and then analyzing the response [4]. All systems with detectable faults (e.g., faults which generate errors at the output) are declared defective and discarded. However, in our work we categorize those faulty systems into those exhibiting acceptable and unacceptable faults, and the systems with acceptable faults can be used. Now the target of testing is to check whether a specific system with an unknown fault is acceptable or not. We assume that DCT and PE architectures are given, and the acceptable and unacceptable set of faults (i.e., F_A and F_{NA}) are already determined based on the fault analysis in Section 4.4. For a PE architecture, we only consider the non-sequential PE architecture, because it is (i) more error tolerant, (ii) more efficient for data access due to its parallel architecture, and (iii) faster, compared to sequential PE. Moreover, test vector generation methodologies that we will use, are not applicable to sequential logic. Note that non-sequential architecture PE requires more area complexity (e.g., $O(NlogN)$ times), compared to sequential one,

but this area increase is negligible as compared to the other sub-modules within JPEG encoder.

Recently, new automatic test pattern generation (ATPG) methods for error tolerant systems were developed. These are based on error significance (ES) ATPG [30], error rate (ER) ATPG [44], and the combination of error significance and error rate ATPG [43]. Error significance and error rate are considered as key metrics of error tolerance [43], and are defined as a maximum absolute error and a probability of non-zero errors respectively, at the testing circuit output. Here the level at which errors are measured can be different from the fault analysis case. In the fault analysis, module level errors (e.g., DCT level) are measured, while sub-module level errors (e.g., PE level) are used for the testing (see Fig. 4.2 and Fig. 4.1).

In ES (or ER) testing, if error significance (or error rate) of a target circuit is greater than given threshold, then the chip is categorized as unacceptable and discarded. Thus for each unacceptable fault, ES ATPG generates at least one test vector which will cause errors greater than given error significance threshold, when applied to a circuit containing this fault. Similarly, ER ATPG generates test vectors for each unacceptable fault with given error rate threshold. Also the combination of ER and ES testing can be used to generate tests.

To apply those ATPG methods, we first divide the DCT system into small sub-modules whose size is suitable for ATPG. Note that test vectors for each sub-module can be applied to each module separately. Test results from all submodules are gathered to determine whether a target DCT system is acceptable or not. If one of the submodules has unacceptable faults, then the whole system is unacceptable.

Here we use each PE, with 64-bits input and 19-bits output, as a target sub-module. Then for each PE, we obtain sub-module level error rate (ER)- error significance (ES) plot which is required for ER and ES APTG methods [43].

4.5.1 ER-ES plot and Generation of test vectors

In this section, we explain how to draw ER-ES plot, and how we choose different ATPG methods based on that. Because we already know the acceptibility of each fault from fault analysis, now we need to measure (ER,ES) pair of each fault. Each fault may cause a different (ER,ES) pair, which can be obtained using a simulation or an analytical estimation method. This simulation is performed in a similar way as in Section 4.4, except that output can be measured at sub-module level (i.e., each PE). For each fault, we introduce input which is generated from statistical models, to the DCT system with/without the fault, and measure sub-module level ER and ES data. Here (ER,ES) vaules can be different depending on the type of inputs to the sub-modules, i.e., whether they are non-uniform (which would consider real image characteristics) or uniform (uniform distribution to all possible input bus lines). Using those (ER,ES) pairs, we can obtain ER-ES plots such as that in Fig. 4.9. Given our choice for the set of acceptable and unacceptable faults (F_A and F_{NA}, respectively), different strategies can be devised. For example, if faults within F_{NA} and F_A can be separated by single constant error significance line (e.g., solid $line_1$ in Fig. 4.9), only an ES ATPG method is needed. But if more complex lines (e.g., piecewise constant dotted $line_2$ in Fig. 4.9) are needed, we need a combination of ER and ES ATPG methods. However, we may choose an ES APTG if a sufficient number of acceptable faults are clustered using a single

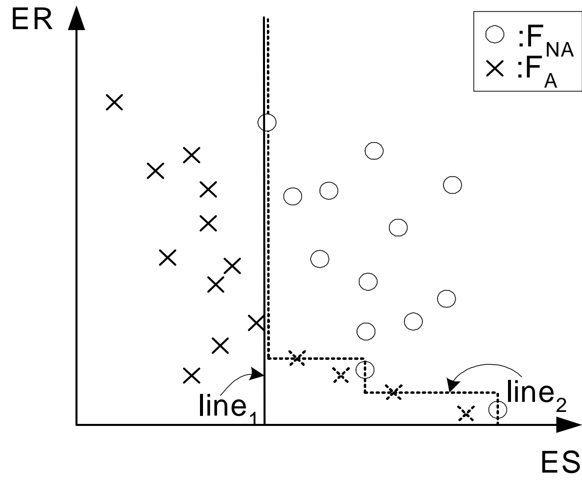


Fig. 4.9: ER-ES plot

error significance threshold. In our work, an ES ATPG method shows sufficiently good testing quality. Additional improvements can be achieved using an ER ATPG (e.g., systems with the dotted faults in Fig. 4.9 can be considered acceptable).

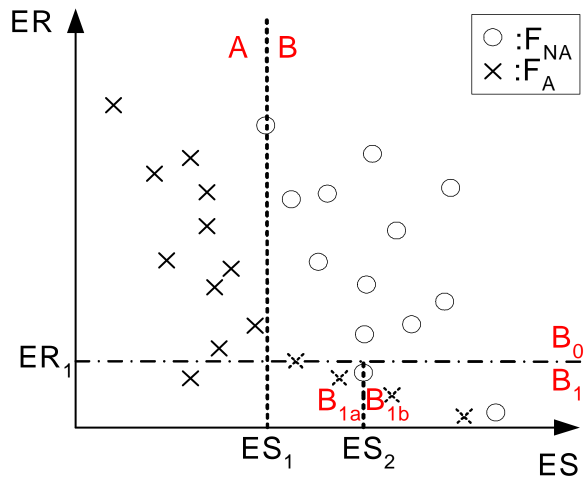


Fig. 4.10: How to decide ER and ES threshold values

For an ES ATPG, we need to decide a constant error significance threshold (e.g., ES_1 in Fig. 4.10). This can be done as follows.

$$ES_i = max(ES \in ES_{NA}), \qquad (4.11)$$

where ES_{NA} is the set of ES values of unacceptable faults within specific region (e.g., ES_1 using ES values of unacceptable faults in whole region in Fig. 4.10). This approach assures that no unacceptable faults are considered to be acceptable after test operations, thus we can declare all systems which pass this test to be acceptable, e.g., systems with faults in region **A** of Fig. 4.10. However, there may be remaining systems with acceptable faults which are discarded in this test, e.g., systems with acceptable faults in region **B** of Fig. 4.10.

Thus, for more accurate testing, we can perform several pairs of ER and ES ATPG tests, which leads to the combination of ER-ES APTG methods. For the ER ATPG, we select an error rate threshold (e.g., ER_1 in Fig. 4.10) as follows.

$$ER_i = max(ER \in ER_A), \qquad (4.12)$$

where ER_A is the set of ER values of acceptable faults within specific region (e.g., ER_1 using ER values of acceptable faults in region **B** of Fig. 4.10). Using this test, we can declare systems which are in region **B** and satisfy $ER > ER_i$, e.g., systems in $\mathbf{B_0}$, as surely unacceptable. And then using ES test (e.g., using ES_2), we can obtain another fraction of systems with acceptable faults (e.g., systems with faults in $\mathbf{B_{1a}}$). Repeated pairs of ER and ES tests generates more complex piecewise constant line (e.g., $line_2$ of Fig. 4.9) which may increase the accuracy of the tests.

As mentioned above, the ER-ES plot can also be obtained using an analytical approach which is described in Algorithm 3.

Algorithm 3 Algorithm to estimate an ER-ES plot

Step 1. For each SSAk ($k = 0, 1$) fault f_i at the r-th bit of specific bus of PE architecture, derive the corresponding distribution of input to that bus lines.
Step 2. Derive p_k=$P(r - th \;\; bit = k)$ based on the distribution, as follows.

$$p_k = P(m2^r + k2^{r-1} \leq X < m2^r + 2^{r-1} + k2^{r-1}), \tag{4.13}$$

where X is the input to that bus and $m \geq 0$ (refer to Fig. 4.11).
Step 3. If $1 - p_k = 0$, then ER=0 and ES=0. Otherwise ER=$1 - p_k$ and ES=2^r

---

For example, for a specific input bus in the PE architecture, e.g., bus for X_1 in Fig. 4.2, X_1 can be represented as a linear combination of $Y(u, v)$s, where each of those is modeled as independent Gaussian. Therefore X_1 can be modeled as a Gaussian as well. Then the p_k and the corresponding ER and ES can be easily obtained as in Algorithm 3.

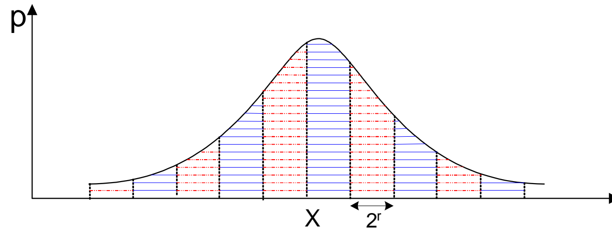
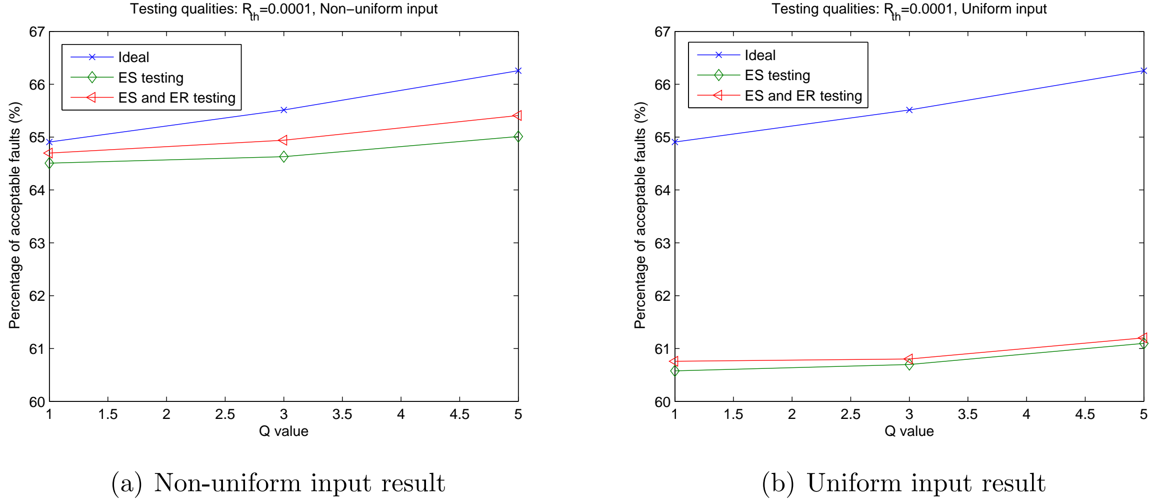


Fig. 4.11: p_k computation; dotted: p_0, solid: p_1

4.5.2 Comparison of Testing qualities and costs

In this section, we estimate testing qualities and testing costs for different test vector generation methods. We provide several comparisons: i) ER-ES plot using uniform vs. non-uniform input, ii) ES ATPG only vs. ER-ES ATPGs combined,

and iii) using estimated vs. simulated ER-ES plot. These tests are applied to the non-sequential PE architecture.



(a) Non-uniform input result

(b) Uniform input result

Fig. 4.12: The percentage of acceptable faults using Ideal, ES, and ER&ES tests with both uniform (U) and non-uniform (NU) input sets; for $R_{th} = 0.0001$ and a non-sequential PE

In Fig. 4.12, an "Ideal" category shows the percentage of acceptable faults using fault analysis. The other categories show the percentage of acceptable faults we can detect using specific testing methods. Our testing shows good testing quality (e.g., over 98% for non-uniform input cases). Also it is interesting to see that error significance testing shows fairly competitive testing quality compared to the combination of ER-ES testings (e.g., less than 1% difference for both uniform and non-uniform input cases). Testing methods using non-uniform input show better performance than the uniform input case, because our fault analysis is based on the non-uniform input which considers the characteristics of images. When we consider the results using estimated ER-ES plots, they are almost the same as the results shown in Fig. 4.12 (e.g., less than 0.4% difference for all cases).

Testing costs mainly depend on the costs of generating and applying test vectors. It has been shown that test generation costs of ER and ES ATPGs are reasonable as compared to traditional tests (e.g., in most cases, test generation costs of an error significance and error rate ATPG methods stay under $2 \sim 3$ times classical ATPG costs) [30, 43]. Test application costs are proportional to the test set size (i.e., number of test vectors). For classical ATPG, the number of test vectors is $O(N_F)$ where N_F is the number of faults within fault space F. In our ER and ES tests, number of test vectors is also $O(N_F)$. In our DCT system, $N_F = 11552$ for the non-sequential PE architecture, and $N_F = 2912$ for the sequential PE archtiecture. Note that the actual test set size of an ER (or ES) ATPG is less than twice that of a classical ATPG [30, 43].

Thus test application costs of ER and ES ATPGs are also similar to the classical ATPG, and with reasonable testing costs, we obtain significant portion of acceptable systems, which leads to better yield rate.

4.5.3 Increases in Yield Rate

Based on a simple probability model, we can estimate the effect of error tolerance on the yield rate. We employ the Poisson statistics in our analysis [7]. Let us assume that faults are distributed uniformly on a wafer which is used to fabricate a chip. Also let us assume that SSA0 and SSA1 are equi-probable, that is, $P_{SSA0} = P_{SSA1}$ where P_{SSA0} and P_{SSA1} are the probability of SSA0 and SSA1, respectively. In this case, the probability of having k faults is given by the Poisson probability mass function

$$P_\lambda(Z = k) = \frac{e^{-\lambda}\lambda^k}{k!}, \tag{4.14}$$

where Z is a random variable which represents the number of faults within some area of a wafer. In the traditional testing, only system without faults contribute to the yield rate (Y_r), which can be measured as $P(Z = 0) = e^{-\lambda}$. But our new testing scheme allows systems with a single "acceptable" fault to be considered acceptable. Thus yield rate is increased by $P(Z = 1)P_A$ where P_A is the percentage of acceptable single faults, and the new yield rate (Y_r^{new}) is measured as $P(Z = 1)P_A + P(Z = 0)$. Note that for given yield rate ($Y_r = e^{-\lambda}$), we can compute λ and $P(Z = 1)$.

In Table 4.3, we provide results for Y_r^{new} vs. Y_r for the DCT circuit we considered when reasonable parameters (i.e., Q and R_{th}) are applied. Our result shows significant yield rate increases. For example, in the early stage of a chip production which has very small traditional yield rate, e.g., 20%, we can have an additional 20.8% acceptable chips, which means doubling the yield rate (refer to Table 4.3). Also in a very late stage, e.g., 80% traditional yield rate, we still have additional 11.5% more yield rate.

Y_r	Y_r^{new}	$\frac{Y_r^{new}}{Y_r}$
20	40.8	2.04
30	53.3	1.78
40	63.7	1.59
50	72.4	1.44
60	79.8	1.33
70	86.1	1.23
80	91.5	1.14
90	96.1	1.07

Tab. 4.3: Original yield rate (Y_r) vs. new yield rate (Y_r^{new}) in % due to tests considering error tolerance for the DCT system with a non-sequential PE; $Q = 1$ and $R_{th} = 0.0001$

4.6 Conclusion

We presented a general framework for analyzing linear transforms followed by quantization. We estimated faults' impact on the system, by analyzing quantization block and linear transform block separately. Faults in the linear transform are modeled as errors added to the output of transform. More specifically we chose the DCT as a transform of interest, and introduced metrics and perceptual thresholds for acceptable degradation, which enables categorizing chips into acceptable and unacceptable. Using these, we proposed a general fault analysis for this system. Our results showed that a significant fraction (e.g., over 50% in some cases) of interconnection faults within a DCT system are acceptable in a JPEG encoder operating at any of its available compression rates. We also proposed a systematic testing method which uses a composite of error rate and error significance metrics of submodules. Our results showed that our testing method provides significant testing coverage (i.e., over 98%), and can achieve significant increases in yield with relatively low testing costs.

Chapter 5

Conclusion and Future Work

In this thesis, we propose a system-level error tolerance scheme for the key components in multimedia compression systems, i) a linear transform combined (e.g., DCT) with quantization and ii) a motion estimation. In both components, we first presented general frameworks for estimating faults' impact on the system, and evaluated the impact using those. Note that we dealt with SSA faults in the DCT, and soft errors due to voltage over scaling in the ME. In a DCT system, we show that a significant fraction (e.g., over 50% in some cases) of interconnection faults within a DCT system are acceptable in a JPEG encoder operating at any of its available compression rates. To exploit that error tolerance, we introduced a systematic testing which uses error rate and error significance as key metrics. This shows quite accurate testing quality and leads to significant increases in yield (i.e., as much as double the original yield in some cases) with relatively low testing costs. In a ME system, we first show that the input voltage to the ME can be reduced under normal operating voltage because of the tolerance to soft errors in ME. Then we model system level performance degradation as a function of input voltage and input characteristics, which enables us to reduce power consumption

in ME (e.g., 37% power saving). We also generalize this work by introducing a framework to use multiple low complexity metrics for ME. This framework considers the trade off between complexity and ME system performance degradation. Within this framework, we discuss (i) desirable characteristics for these metrics, (ii) how to estimate complexity-degradation characteristics, and (iii) how to combine multiple metric computations to estimate best motion vector (MV). As a case study, we apply above framework to the combination of voltage over scaled (VOS) and sub-sampled (SS) metrics as well as newly introduced two SSVOS metrics. We introduce a new two metric system whose performance is better than combination of VOS and SS metrics (87% power saving with less than 2% degradation).

As future work, we first can extend the DCT approach to other transforms such as wavelet transforms. We can also pursue low power DCT considering the tolerance to soft errors in a DCT cases. As for ME, we can perform more research on error modeling for other low complexity metric computations, and we can analytically investigate the effect of different ME algorithms.

Bibliography

[1] ISO/IEC JTC1/SC29/WG11 N2202 Committee Draft., 1998.

[2] Information technology-JPEG 2000-image coding system-part 1: Core coding system., 2000.

[3] Draft ITU-T recommendation and final draft international standard of joint video specification (ITU-T Rec. H.264/ISO/IEC 14 496-10 AVC in Joint Video Team(JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050, 2003.

[4] M. Abramovici, M. A. Breuer, and A. D. Friedman. *Digital Systems Testing and Testable Design.* The Institute of Electrical and Electronics Engineering, Inc., New York, 1990.

[5] A. J. J. Ahumada and A. Watson. An improved detection model for DCT coefficient quantization. In *Human Vision, Visual Processing, and Digital Display IV. Allebach ed., SPIE*, 1993.

[6] G. Bjontegaard. Calculation of average PSNR differences between RD-curves. *ITU-T Q.6/SG16 VCEG, VCEG-M33*, April 2001.

[7] P. J. Bonk, M. R. Gupta, R. A. Hamilton, and A. V. S. Satya. *Microelectronics Manufacturing Diagnostics Handbook.* Kluwer Academic Publishers, New York, 1993.

[8] M. A. Breuer, S. K. Gupta, and T. M. Mak. Defect and error tolerance in the presence of massive numbers of defects. *IEEE Design & Test of Comp.*, 21:216–227, May–June 2004.

[9] A. P. Chandrakasan and R. W. Brodersen. Minimizing power consumption in digital CMOS circuits. *Proc. of IEEE*, 83(4):498–523, Apr. 1995.

[10] H. Cheong, I. Chong, and A. Ortega. Computation error tolerance in motion estimation algorithms. In *IEEE International Conference on Image Processing, ICIP'06*, Oct. 2006.

[11] H. Y. Cheong and A. Ortega. System level fault tolerant motion estimation algorithms & techniques. Technical report, SIPI, Univ. of Southern California, 2006.

[12] H.-Y. Cheong and A. M. Tourapis. Fast motion estimation within the H.264 codec. In *Proc. of the Int. Conf. on MultiMedia and Expo (ICME-2003)*, July 2003.

[13] T. Chiang and Y. Q. Zhang. A new rate control scheme using quadratic rate distortion model. *IEEE Trans. Circuits Syst. Video Technol.*, 7(1):246–250, Feb. 1997.

[14] I. Chong and A. Ortega. Hardware testing for error tolerant multimedia compression based on linear transforms. In *Proc. of IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, DFT'05*, pages 523–534, 2005.

[15] I. Chong and A. Ortega. Dynamic voltage scaling algorithms for power constrained motion estimation. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[16] I. Chong and A. Ortega. Power efficient motion estimation using multiple imprecise metric computations. In *International Conference on Multimedia and Expo (ICME)*, 2007.

[17] H. Chung. *Complexity Scalable and Robust Motion Estimation for Video Compression*. PhD thesis, Univ. Southern California, 2007.

[18] H. Chung and A. Ortega. System level fault tolerance for motion estimation. Technical Report USC-SIPI354, Signal and Image Processing Institute, Univ. of Southern California, 2002.

[19] E. Debes. Recent changes and future trends in general purpose processor architectures to support image and video applications. *Proc. of the International Conference on Image Processing*, 3:85–88, 2003.

[20] M. H. DeGroot and M. J. Schervish. *Probability and Statistics*. Addison Wesley.

[21] F. Dufaux and F. Moscheni. Motion estimation techniques for digital TV: A review and a new contribution. *Proc. of the IEEE*, (6):858–876, June 1995.

[22] M. A. Elgamel, A. M. Shams, and M. A. Bayoumi. A comparative analysis for low power motion estimation VLSI architectures. In *IEEE Workshop on Signal Processing*, 2000.

[23] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge. Razor: A low-power pipelined based on circuit-level timing speculation. *MICRO-36*, Dec. 2003.

[24] A. Eskicioglu. Quality measurement for monochrome compressed images in the past 25 years. In *Proc. of the International Conference on Acoustics, Speech and Signal Processing*, pages 1907–1910, 2000.

[25] B. Girod. *What's wrong with mean-squared error*. MIT Press, United States, 1993.

[26] M. Gossel and E. S. Sogomonyan. New totally self-checking ripple and carry look-ahead adders. In *3rd Int. On-line Testing Workshop*, 1997.

[27] Z.-L. He, K.-K. Chan, C.-Y. Tsui, and M. L. Liou. Low power motion estimation design using adaptive pixel truncation. In *Proc. of Int. Symp. Low Power Electronics and Design*, pages 167–172, 1997.

[28] R. Hedge and N. R. Shanbhag. Soft digital signal processing. *IEEE Trans. on VLSI systems*, 9(6), 2001.

[29] N. Jha and S. Gupta. *Testing of Digital Systems*. Cambridge University Press, United Kingdom, 2003.

[30] Z. Jiang and S. Gupta. An ATPG for threshold testing: Obtaining acceptable yield in future processes. In *International Test Conference*, 2002.

[31] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro. Motion-compensated interframe coding for video conferencing. In *IEEE NTC*, pages 531–534, 1981.

[32] I. Koren and Z. Koren. Defect tolerant VLSI circuits: Techniques and yield analysis. In *Proceedings of the IEEE, Vol. 86*, pages 1817–1836, San Francisco, CA, Sept. 1998.

[33] P. Kuhn. *Algorithms, complexity analysis and VLSI architectures for MPEG-4 motion estimation*. Kluwer Academic Publishers, Boston, 1999.

[34] K. Lengwehasatit and A. Ortega. Probabilistic partial-distance fast matching algorithms for motion estimation. *IEEE Trans. Circuits Syst. Video Technol.*, 11(2):139–152, Feb. 2001.

[35] R. E. Lyions and W. Vanderkulk. The use of triple modular redundancy to improve computer reliability. Technical report, IBM J. Res. Develop., 1962.

[36] J. L. Mitchell and W. B. Pennebaker. *JPEG Still image data compression standard*. Van Nostrand Reinhold, New York, 1993.

[37] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall. *MPEG Video Compression Standard*. Chapman and Hall, New York, 1997.

[38] T. Mudge. Power: A first class design constraint. *IEEE Computer*, 34(4):52–57, April 2001.

[39] P. Pirsch, N. Demassieux, and W. Gehrke. VLSI architectures for video compression-a survey. In *Proc. IEEE*, volume 83(2), pages 220–246, Feb. 1995.

[40] K. R. Rao and P. Yip. *Discrete Cosine Transform.* Academic Press, United States, 1990.

[41] R. Rosenholtz and A. B. Watson. Perceptual adaptive JPEG coding. In *Proceedings of ICIP*, volume 1, pages 901–904, 1996.

[42] K. Sauer and B. Schwartz. Efficient motion estimation using integral projections. *IEEE Trans. on Circuits and Systems for Video Technology*, 6(5):513–518, Dec. 1996.

[43] S. Shahidi. *Error-rate testing to improve yield for error tolerant applications.* PhD thesis, Univ. Southern California, 2008.

[44] S. Shahidi and S. Gupta. Multi-vector tests: A path to perfect error-rate testing. In *Proc. of Design, Automation and Test in Europe*, 2008.

[45] K. L. Shepard and V. Narayanan. Noise in deep submicron digital design. In *ICCAD*, pages 524–531, Nov. 1996.

[46] F. Shih. High performance self-checking adder for VLSI processor. In *IEEE Custom Integrated Circuits Conference*, 1991.

[47] O. Stern and H. J. Wunderlich. Simulation results of an efficient defect analysis procedure. In *Proc. IEEE Int. Test Conf. on TEST: The Next 25 Years*, pages 729–738, Oct. 1994.

[48] P. Tseng, Y. Chang, Y. Huang, H. Fang, C. Huang, and L. Chen. Advances in hardware architectures for image and video coding- a survey. *Proc. of the IEEE*, (1):184–197, Jan. 2005.

[49] G. V. Varatkar and N. R. Shanbhag. Energy-efficient motion estimation using error-tolerance. In *International Symposium on Low Power Electronics and Design (ISLPED)*, 2006.

[50] G. V. Varatkar and N. R. Shanbhag. Error-resilient motion estimation architecture. *Transactions on VLSI Systems*, January 2008.

[51] A. B. Watson, J. Hu, and J. F. McGowan. Digital video quality metric based on human vision. In *Journal of Electronic Imaging*, volume 10(1), pages 20–29, 2001.

[52] T. Wiegand and B. Girod. Lagrange multiplier selection in hybrid video coder control. In *Proc. of the International Conference on Image Processing (ICIP'01)*, volume 3, pages 542–545, 2001.

Appendix

Advantages of positive biased estimator

In this appendix we prove that a positive biased estimator of each SAD_i is an efficient tool to minimize degradation. We make the following assumptions. First, we use general error models of metric computations and the MV estimation approach in Fig. 3.3 (b), without specifying any estimator of each SAD_i. Second, we assume that unknown deterministic parameters $SAD_1, ..., SAD_N$ are in ascending order ($SAD_1 < ... < SAD_N$). Third, as in Section 3.3, we assume that our metrics have small estimation errors for SAD_1 as compared to other SADs, i.e., errors tend to be smaller for small magnitude SADs.

For each candidate vector i, $\widehat{SAD_i}$ is obtained from M computations, which has estimation error $\widehat{E_i} = \widehat{SAD_i} - SAD_i$. That error as well as $\widehat{SAD_i}$ have distributions which depend on the estimator and error models of SAD computations. Define conditional probability mass function (pmf) of $\widehat{SAD_i}$ as $f_i(y_i) = P(\widehat{SAD_i} = y_i|SAD_i)$ and $F_i(y_i)$.

Now E_{SAD} can be computed as follows.

$$E_{SAD} = \sum_{j=2}^{N} (SAD_j - SAD_1) P(\hat{I} = j), \tag{5.1}$$

where $P(\hat{I} = j)$ can be defined as

$$P(\hat{I} = j) = \int_{-\infty}^{\infty} \int_{-\infty}^{x_1} \int_{x_j}^{\infty} ... \int_{x_j}^{\infty} f_1(y_1) f_j(y_j) f_2(y_2) ... f_N(y_N) dy_N ... dy_2 dy_j dy_1. \tag{5.2}$$

And $P(\hat{I} = j)$ can be written as

$$P(\hat{I} = j) \approx F_j(\gamma) \prod_{i=2, i \neq j}^{N} (1 - F_i(\gamma)), \quad \gamma = E(\widehat{SAD_1}) \tag{5.3}$$

Here it is clear that minimizing $P(\hat{I} = j)$ for each j leads to minimizing E_{SAD}, where this approach for "each j" is reasonable because a SAD estimator in the current framework is for "each j". Also it is shown that a decrease in $F_j(\gamma)$ leads to a decrease in $P(\hat{I} = j)$ (See below for details). If we decrease each $F_j(\gamma)$ by factor of $\alpha < 1$, $P(\hat{I} = j)$ can be rewritten as follows.

$$P(\hat{I} = j) \approx \alpha F_j(\gamma) \prod_{i=2, i \neq j}^{N} (1 - \alpha F_i(\gamma)), \tag{5.4}$$

which monotonically decreases as a function of α, if following holds:

$$\sum_{k=2, k \neq j}^{N} F_k(\gamma) < \frac{1}{2\alpha_{max}}, \alpha_{max} = max(\alpha) < 1, \tag{5.5}$$

where it usually holds for the current choice of ME algorithm, i.e., EPZS has very small N.

In summary, small $F_j(\gamma)$ for each $j > 1$ is a necessary condition for minimizing E_{SAD}. The same conclusion also can derived in the sense of minimizing P_{err} which is defined as probability of $\hat{I} \neq I$ and can be written as follows.

$$P_{err} = P(\hat{I} \neq I) \approx 1 - \prod_{j=2}^{N} (1 - F_j(\gamma)) \tag{5.6}$$

Based on definition of P_{err}, it is obvious that the same condition holds as a necessary condition for minimizing P_{err}. Thus we need to find combination of metrics with a proper estimator which minimizes $F_j(\gamma)$.

To minimize $F_j(\gamma)$, small $\gamma = E(\widehat{SAD_1})$ and small $F_j(x), j > 1$ are needed for small x. Clearly, a positively biased estimator for $\widehat{SAD_j}$ has small $F_j(x)$ for small x, because $f_j(x)$ should be biased to bigger x values. A negatively biased estimator for $\widehat{SAD_1}$ ensures small $E(\widehat{SAD_1})$. But the same estimator should be applied to all SAD_is. Thus one reasonable solution is a positively biased estimator but with $\widehat{SAD_1} \approx SAD_1$. Note that we assume our choice of metrics already have possibility of small estimation errors for small SAD_i, thus estimators need to exploit that possibility.