Vol.2019-ARC-235 No.50
Vol.2019-SLDM-187 No.50
Vol.2019-EMB-50 No.50
2019/3/18

BER測定を用いたストカスティック数の誤り訂正

石川 遼太[1]　多和田 雅師[1]　柳澤 政生[1]　戸川 望[1]

概要：電子回路では熱雑音など，防ぐことのできないノイズが発生する．ノイズにより 2 進数によって表現された信号では，上位ビットが反転すると値が大きく増減する．一方，各ビットが同じ重みを持つストカスティック数を用いると，どのビットが反転しても値の増減は一定である．そのため，ノイズによる影響が小さいストカスティック数を用いた計算手法であるストカスティックコンピューティングが注目を集めている．ストカスティックコンピューティングはエラーに対する耐性を持つが，ノイズが乗ると 2 進数と同様にビット列を復元することができない．ここで，出力されたストカスティック数の値とビットエラーレートから元のストカスティック数の値を推測することを考える．本稿では，BER (Bit Error Rate) を測定し適切なフィルタをかけることで，ストカスティック数の誤り訂正する手法を提案する．実験結果から 22%以上の確率でビットの反転が起きる環境では，この手法によって既存の誤り訂正手法よりも PSNR (Peak Signal-to-Noise Ratio) が向上することを確認した．

1. はじめに

1.1 研究背景

近年，演算対象の情報量の増加と演算内容の複雑化の要求に伴い半導体製造プロセスの微細化が進んでいる．半導体プロセスが微細化すると放射線や配線内の熱雑音による影響が相対的に大きくなり, 配線内信号にビット誤りが発生する確率 (Bit error rate, BER) が高まる．ビット誤りを訂正する一般的な手法として, 信号を符号化して誤り耐性を高める誤り訂正符号 [1], [2], [3] が存在する．これらの手法は BER が高いと誤り訂正能力が著しく低くなってしまうという問題がある．一方で画像処理や音声処理を始めとする一部の計算分野では常に正確な演算が求められる訳ではなく, 人間の認識の範囲外での誤りは許容される．BER が高い環境下では厳密に正確な演算を保障することは現実的ではなく, ある程度の演算誤りを許容した上で誤り量を削減することが重要となる．

誤りに耐性を持つ計算手法としてストカスティックコンピューティング (Stochastic computing, SC)[4] が存在する．SC ではストカスティック数 (Stochastic number, SN) を用いて計算を行なう．SN を構成するビット列はその重みが一定であるという性質を持ち, ビット誤りに対して誤り量が一定となる．2 進数を構成するビット列はその重みがビット位置により異なり, ビット誤りに対して誤り量がばらつく．SN は 2 進数を比較すると誤り量の最大値が相対的に小さくなるため, 誤りに耐性を持つビットエンコーディングである．また, SC の演算は SN を構成するビット列の各ビットが独立に演算結果に寄与するため, 誤りの伝搬が起こりにくい．SC は誤りに耐性を持つが, 誤りが発生した後で誤り量そのものを削減する誤り訂正ができない．

[1] 早稲田大学大学院 基幹理工学研究科 情報理工・情報通信専攻

1.2 提案手法

本稿では SC を使用した誤り訂正可能な演算システムを構築する．SN の全てのビットは同じ重みを持つという構成から, BER が時間によらず一定の環境では BER の値が推定できれば誤りの乗っていない正確な SN の値を計算できる性質を証明する．証明された性質をもとに誤り訂正可能な演算システム *SN-DN* を提案する．SC で用いられる演算対象の SN 集合に BER 測定用の SN としてキャリブレーション SN を追加することを考える．キャリブレーション SN に発生した誤り量から BER の値を推定し, 演算結果の各 SN に発生した誤りを訂正する．画像処理システムに *SN-DN* を適用したシミュレーション実験により画像の Peak signal-to-noise ratio (PSNR) を測定し, BER の大きい環境では既存の誤り訂正手法よりも提案手法の誤り量が小さくなることを確認する．

1.3 本稿の貢献点

本稿の貢献点は以下の通りである．

(1) SC において BER の値が推定できれば誤りの乗っていない値を計算できることを証明する．
(2) 誤り訂正手法 *SN-DN* を提案する．
(3) BER が 22%以上のとき *SN-DN* を画像の誤り訂正に

© 2019 Information Processing Society of Japan

Vol.2019-ARC-235 No.50
Vol.2019-SLDM-187 No.50
Vol.2019-EMB-50 No.50
2019/3/18

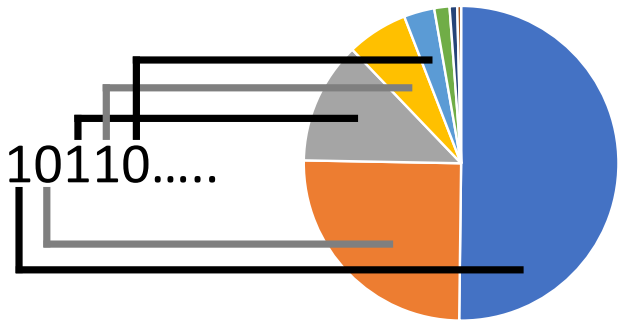


図 1　2 進数の各ビットの重み.

用いると，PSNR が既存手法と比べて大きくなることを確認する．

1.4　本稿の構成

本稿の構成は以下の通りである．2 章では誤りモデルと既存の誤り訂正を議論する．3 章ではストカスティック数 (SN) を紹介し，2 進数とともにその誤り耐性を議論する．4 章では *SN-DN* という誤り訂正手法を提案し，その性質を議論する．5 章では実験の評価を通して *SN-DN* の有用性を示す．6 章では本稿をまとめる．

2.　誤りモデルと既存手法

2.1　誤りモデル

本稿で対象とする誤りモデルとして以下の条件を想定する．

(1) 回路内で演算対象となるデータを通信するときに誤りが発生
(2) 誤りの発生確率は時間経過によらない BER に則り，ビットごとに独立
(3) 誤りの種類はビット反転
(4) 符号化処理・復号処理において誤りはなし

2.2　既存手法

データにビット反転誤りが発生するときに使用される誤り訂正符号が存在する．特に回路内で演算ごとに符号化・復号を繰り返すためブロック誤り訂正符号を考える．既存の誤り訂正符号としてハミング符号 [1]，ゴレイ符号 [2]，ポーラ符号 [3] がある．これらの手法は BER が高いと誤り訂正能力が低くなってしまうという問題がある．BER が高いときに誤り訂正能力が高くなる誤り訂正符号として反復符号 (Repetition Code)[5] が存在する．

反復符号は符号化処理と復号処理によって構成される．符号化処理は，データの 1 ビットを N ビットに複製して全体で N 倍に冗長化することで符号化する．復号処理は，データのうち N ビットに複製されたもとの 1 ビット多数決法により推定して誤り訂正する．この手法を本稿では N-RC と呼ぶ．

N-RC を用いた誤り訂正システムを図 2 に示す．図 2 では画像を (1) 符号化処理し $N = 4$ 倍に冗長化し符号語とする．得られた符号化されたデータを (2) 送信し，その過程で (3) エラーが発生する．そのため (4) 受信したデータは誤りの乗ったデータとなり，(5) 復号処理によりもとの画像へ誤り訂正される．

3.　ビットエンコーディングと誤り耐性

3.1　2 進数によるビットエンコーディング

演算システムの演算対象のデータを誤り訂正符号化されていない 2 進数で構成することを考える．n ビットの 2 進数 bin の定義域は $0 \leq bin \leq 2^n - 1$ であるから，定義域の最大値に対する i 桁目のビットの重みの比率は図 1 の通り，

$$\frac{2^{i-1}}{2^n - 1} \tag{1}$$

となる．ただし，$1 \leq i \leq n$ とする．ここから，誤るビットによっては定義域の最大値の約半分の誤差を生むことがわかる．

また，ビット反転誤りによる誤り量は誤るビット位置によって異るため，誤った値から元の値を推測することはできない．そのため N-RC のようなブロック誤り訂正符号で符号化・復号し，元のビット列に戻すことでしか誤り訂正を行えない．

2 進数を N-RC を用いて誤り訂正することを考える．本稿では各ピクセルの各色を 0–255 とし，8 ビットの 2 進数で表現する．そのため，512×512 ピクセルの画像の通信には $N = 16$ とすると，

$$8[bit/color \cdot pixel] \times 3[color] \times (512 \times 512)[pixel] \times 16 = 100,663,296[bit] \tag{2}$$

必要になる．冗長化を $N = 16$ 倍としているのは，1 つのデータあたりの情報量を後述する提案手法と合わせるためである．

3.2　ストカスティック数によるビットエンコーディング

ストカスティックコンピューティング (SC) では，ストカスティック数 (SN) を用いて計算する．SN は，各ビットが 0 と 1 で構成される任意の長さのビット列である．SN x について，ビット長を $|x|$，i 番目のビットを x_i と表す．SN x のビット列の内の 1 の出現回数を S_x とすると，x の値 V_x は 以下の式 (3) で定義される．

$$V_x = P_x = S_x/|x| \tag{3}$$

ここで，P_x は x のビット列中の 1 の出現頻度であり，$0 \leq P_x \leq 1$ が成り立つ．例えば，$x = 01000100$ のとき，その値は $V_x = 0.25$ となる．このように $V_x = P_x$ を満たす表現方法を単極表現と呼ぶ．単極表現では，$0 \leq V_x \leq 1$ が成り立つ．単極表現の他にも両極表現 ($V_x = 2 \times P_x - 1$ で

© 2019 Information Processing Society of Japan

Vol.2019-ARC-235 No.50
Vol.2019-SLDM-187 No.50
Vol.2019-EMB-50 No.50
2019/3/18

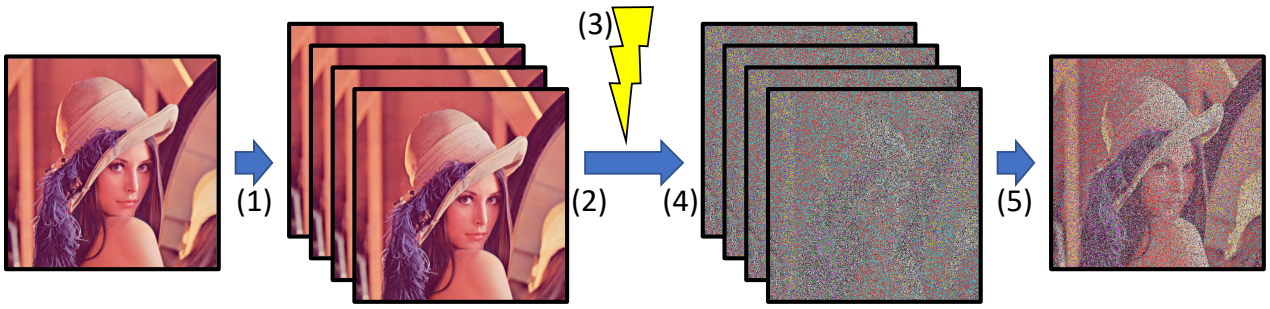


図 2　反復符号 (N-RC) による誤り訂正 ($N = 4$)[5].

定義され，$-1 \leq V_x \leq 1$ が成り立つ) など，SN の表現方法は多数あるが，本稿では，単極表現のみを取り扱う．

SC では論理回路に SN のビット列から 1 ビットずつ順に入力することで算術演算を行う．AND ゲートで乗算を，MUX 回路で加算を，NOT ゲートで減算をそれぞれ実装できる [4].

SN の値はそのビット列中の 1 の割合で定義されるため，ある SN x の各ビットのビット毎の重みはビットによらず全て

$$1/|x| \tag{4}$$

となる．また，BER を p $(0 \leq p \leq 1)$ とすると，ノイズの乗った SN x' の値 $V_{x'}$ の期待値 $E(V_{x'})$ は，

$$\begin{aligned} E(V_{x'}) &= E(P_{x'}) \\ &= P_x - P_x \times p + (1 - P_x) \times p \\ &= (1 - 2p)P_x + p \\ &= (1 - 2p)V_x + p \end{aligned} \tag{5}$$

となる．

4. BER 測定を用いたストカスティック数の誤り訂正

3.2 節より，SN はビット位置によって重みが変らないため誤差のばらつきがほとんど無く，入力 SN の値と BER がわかればノイズの乗った SN の値が推測できることがわかる．逆に，BER が一定でノイズの乗った SN の値と BER が判明すれば，元の SN の値が推測できる．式 (5) の逆関数から，$p \neq 1/2$ のとき元の SN の値の期待値は，

$$\begin{aligned} E(V_x) &= E(P_x) \\ &= \frac{P_{x'} - p}{1 - 2p} \\ &= \frac{1}{1 - 2p} P_{x'} - \frac{p}{1 - 2p} \\ &= \frac{1}{1 - 2p} V_{x'} - \frac{p}{1 - 2p} \end{aligned} \tag{6}$$

となる．式 (5) を式 (6) に代入すると，

$$\begin{aligned} E(V_x) &= \frac{1}{1 - 2p} V_{x'} - \frac{p}{1 - 2p} \\ &= \frac{1}{1 - 2p} \times ((1 - 2p)V_x + p) - \frac{p}{1 - 2p} \\ &= V_x + \frac{p}{1 - 2p} - \frac{p}{1 - 2p} \\ &= V_x \end{aligned} \tag{7}$$

となり，実際に誤りが訂正できていることがわかる．本稿では，式 (6) のフィルタを実装することを考える．$p > 0$ とすると，$-1 \leq 1 - 2p \leq 1$ となるため，式 (6) の傾きの絶対値は 1 より大きくなる．SC では傾きが 1 より大きい一次関数は実装できないため，2 進数による演算でこのフィルタを実装する必要がある．

このフィルタを用いた誤り訂正手法 *SN-DN* (SN based De-Noising) を提案する．*SN-DN* は，ノイズが乗る前に元のデータに BER p を測定するための信号を付加し，データを読み込む際に BER を計算し，それによって誤りを訂正することで，元のデータを復元する．誤り訂正の手順は以下のとおりである．

(1) 入力データを SN に変換
(2) SC による演算処理
(3) BER 検出用の信号 e を付加
(4) データの送信
(5) エラーの発生
(6) データの受信
(7) 受信した SN と BER 測定用の信号を 2 進数に変換
(8) BER 検出用の信号とデータにより誤り訂正

本稿では，画像の誤り訂正を取り扱い，その手法を図 3 に示す．(1) では，データ量が大きくなりすぎるのを防ぐために，各ピクセルの各色を 127 ビットの SN で表現する．今回は (2) の演算は恒等変換とする．BER 検出用の信号にはビット長が 1024 で，値が 0 の SN e を用いる．このような e を用いると，ノイズが乗った V_e が p となる．これらから，512×512 ピクセルの画像の通信には

$$\begin{aligned} 127[bit/color \cdot pixel] \times 3[color] \times (512 \times 512)[pixel] \\ + 1024[bit] = 99,877,888[bit] \end{aligned} \tag{8}$$

必要になることがわかる．

© 2019 Information Processing Society of Japan

Vol.2019-ARC-235 No.50
Vol.2019-SLDM-187 No.50
Vol.2019-EMB-50 No.50
2019/3/18

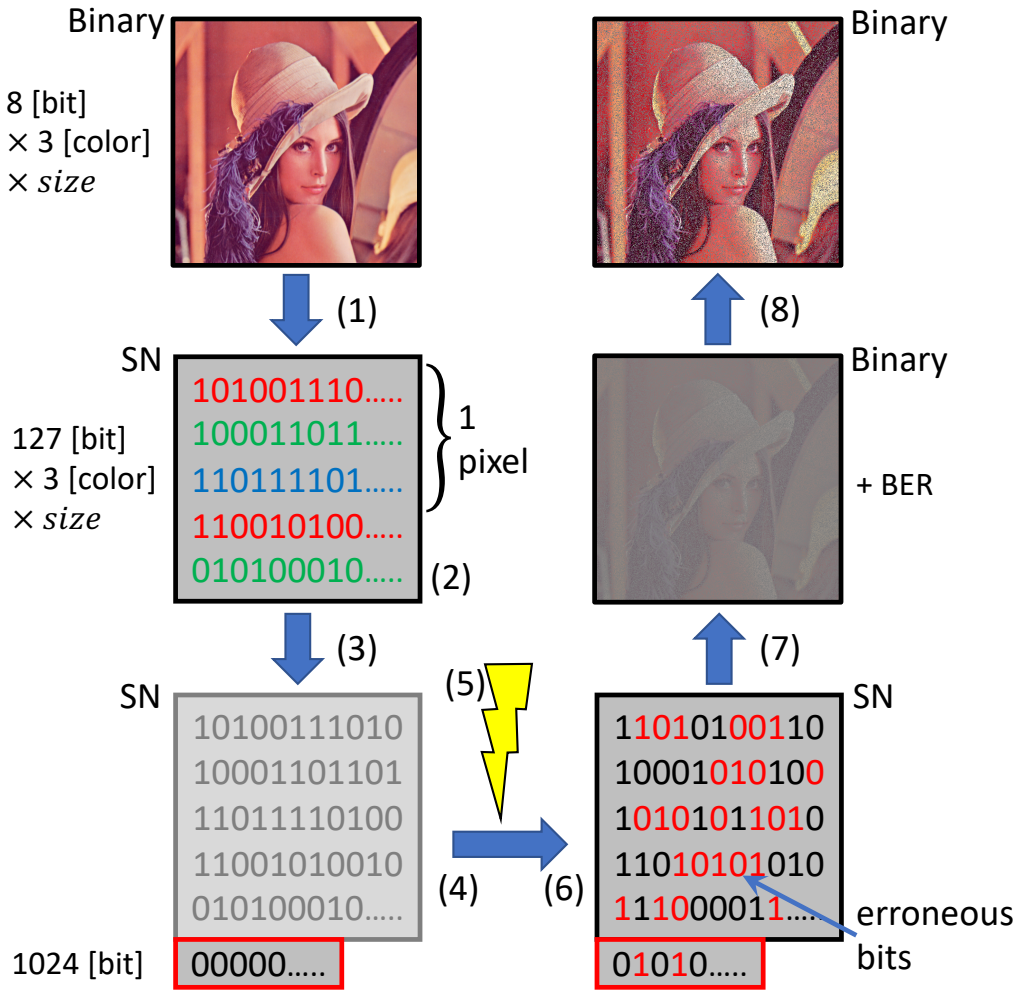


図 3　誤り訂正手法 *SN-DN* の手順.

5. 評価実験

5.1 実験方法

提案する演算システムをシミュレーションし画像に乗った誤り量を測定する. 図 4 の画像を演算システムに入力し, BER の則る誤りを乗せ, その誤りを訂正する. 誤り訂正実装する手法は Binary, 16-RC, SN, *SN-DN* である.

- Binary: 2 進数でビットエンコーディング
- 16-RC: 2.2 節に示す誤り訂正符号
- SN: SN でビットエンコーディング
- *SN-DN*: 4 章の提案手法

であり，以下の条件で実験を行う.

- 実行環境: Python 3.6.3
- 画像フォーマット: 512 × 512 ピクセル，BMP 形式
- ピクセルの各色毎の情報量:
  Binary, 16-RC: 8 ビット
  SN, SN-DN: 127 ビット
- 誤り訂正に用いる情報量 (オーバヘッド):
  16-RC: 元データの情報量の 15 倍の情報量
  SN-DN: 1024 ビット
- 実装手法: Binary, 16-RC, SN, *SN-DN*
- 誤り量: BER が 0–49%
- 評価指標: PSNR



図 4　テスト画像.

なお，PSNR は以下の式で表わされる.

$$PSNR = 20\log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \tag{9}$$

ここで，MSE (Mean Square Error) は以下の式で表わされ,

$$MSE = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{3}(f_{origin}(i,j,k) - f_{actual}(i,j,k))^2 \tag{10}$$

© 2019 Information Processing Society of Japan

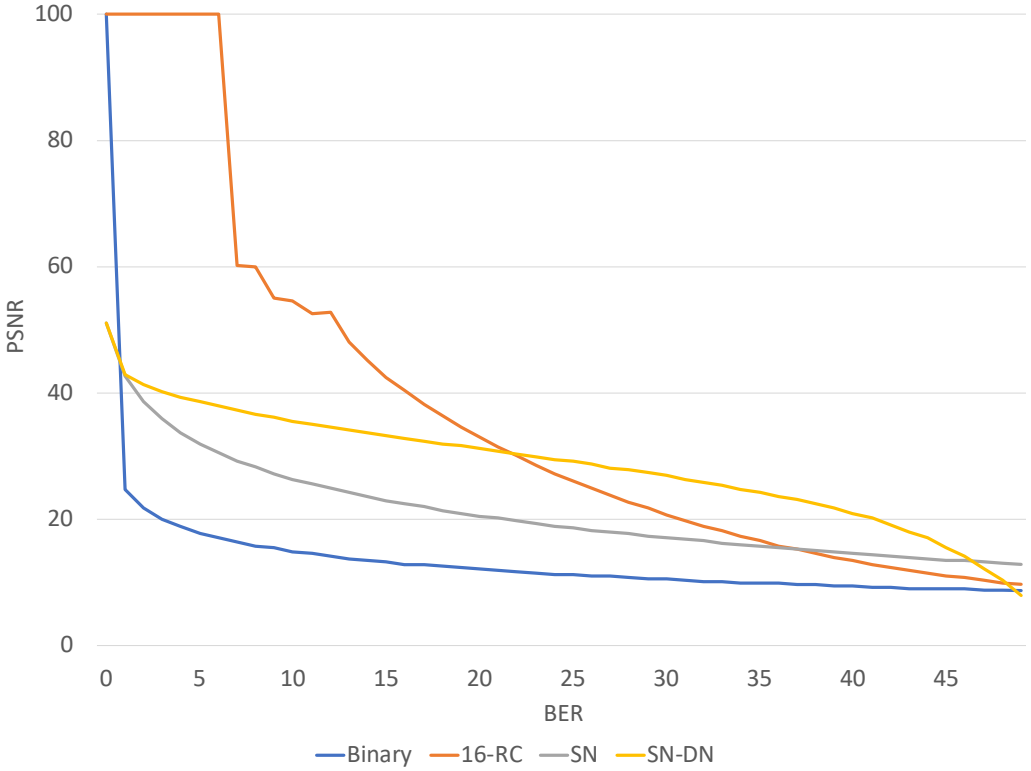


図 5 誤り訂正手法毎の出力の PNSR.

$f_{origin}(i,j,k)$, $f_{actual}(i,j,k)$ はそれぞれ元の画像，出力の画像の (i,j) 番目のピクセル $(1 \leq i \leq m,\ 1 \leq j \leq n)$ の k 番目の色の値を 0 から 255 の値に変換したものである．ただし，m, n はそれぞれ画像の縦と横のピクセル数である $(m = n = 512)$.

5.2 実験結果

図 5 に出力の PSNR を，表 1 に実際の出力画像を示す．PSNR が 100 になっているのは出力画像が元の画像と一致した場合である．SN-DN が SN の PSNR を全体的に向上させたことがわかる．Binary では BER が 0%のとき，元の画像と同一の画像が出力された．一方で，SN を用いる手法では 255 ビットではなく 127 ビットの SN を用いるため，多少の誤差が出た．16-RC では BER が 6%のときまでは完全に誤りを訂正することができたが，その後 PSNR は徐々に低下し 22%のときに SN-DN の PSNR を下回った．

出力画像からも，BER が 25%から 45%のとき 16-RC による誤り訂正よりも SN-DN の方がノイスが少ないことがわかる．16-RC では全体的にノイズが乗るという結果になった．SN-DN では一部のビットの情報が失われてしまい，最小値，最大値，または中間値をとる，という結果になった．

6. おわりに

本稿では，*SN-DN* という誤り訂正手法を提案した．提案手法ではデータを符号化し BER を測定し適切なフィルタをかけることで，SN の誤りを訂正することができる．シミュレーション実験による誤り量の測定では，22%以上 47%未満の確率でビットの反転が起きる環境では提案手法は既存手法よりも誤りを削減できた．

今後は，本稿で提案した *SN-DN* をハードウェアに実装する．また，*SN-DN* を更に拡張してより精度の高い誤り訂正可能な演算システムを設計するとともに，SC による実用的な画像処理回路に実装・検証する．

謝辞

本研究は JSPS 科研費 17K19986 の助成を受けたものです.

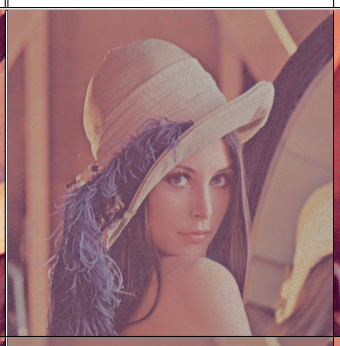
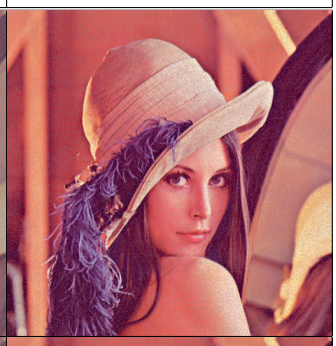
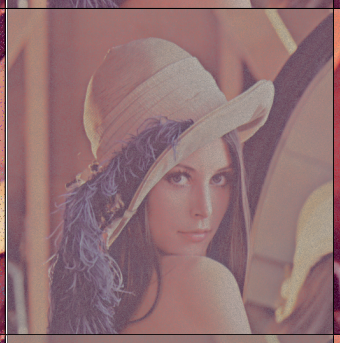
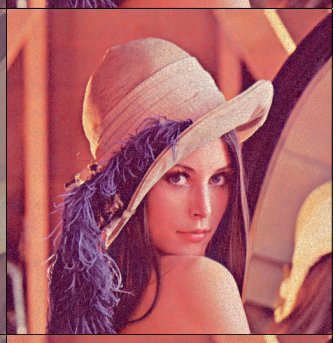
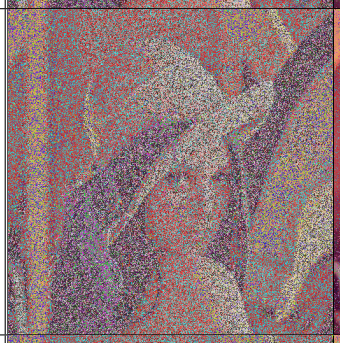
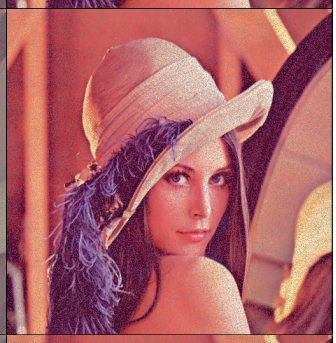
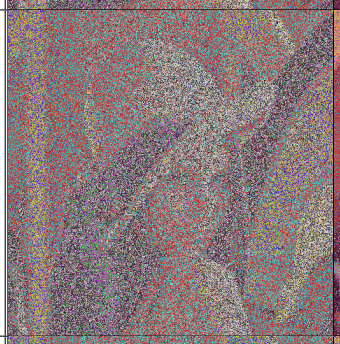
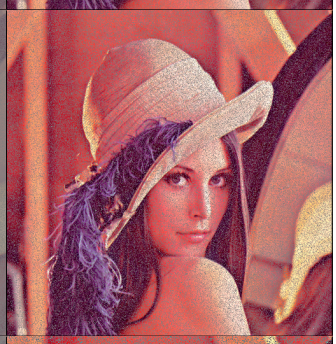
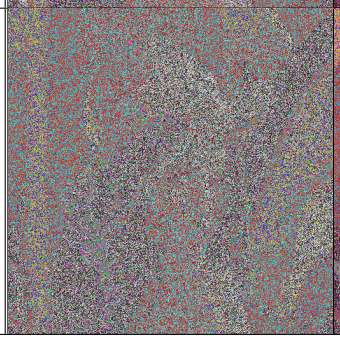
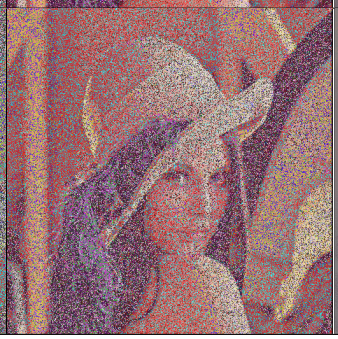
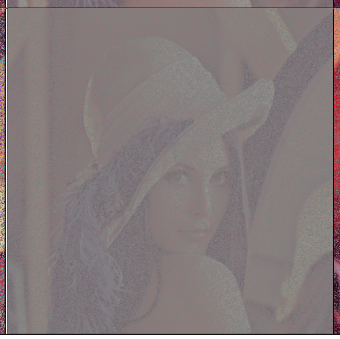
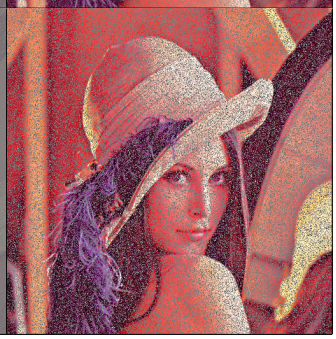
参考文献

[1] R.W. Hamming, “Error detecting and error correcting codes,” The Bell System Technical Journal, vol.29, no.2, pp.147–160, 1950.
[2] M.J.E. Golay, “Notes on digital coding,” Proc. IRE, p.657, 1949.
[3] E. Arikan, “Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels,” IEEE Transactions on Information Theory, vol.55, no.7, pp.3051–3073, 2009.
[4] B.R. Gains, “Stochastic computing,” Proc. Spring Joint Computer Conference, pp.149–156, 1967.
[5] S. Gravano, M.C. Doggett, and P.J. Mcdougall, “Comparison of a cyclic code and a repetition code with the same code rate in the presence of single-bit errors,” International Journal of Electronics, vol.67, pp.495–502, Oct. 1989.

© 2019 Information Processing Society of Japan

Vol.2019-ARC-235 No.50
Vol.2019-SLDM-187 No.50
Vol.2019-EMB-50 No.50
2019/3/18

表 1　BER，誤り訂正手法毎の出力画像.

	Binary	16-RC	SN	SN-DN
25%				
30%				
35%				
40%				
45%				

© 2019 Information Processing Society of Japan