RC-004

ビットパターンドメディアレコーディングのための二重削除／挿入／反転誤り訂正符号

Double Deletion/Insertion/Reversal Error Correcting Code for Bit-Patterned Media Recording

井上雅斗* 金子晴彦 *
Masato Inoue Haruhiko Kaneko

1 はじめに

従来の磁気記憶装置では，熱揺らぎの影響により記録密度が限界に近づいていることが問題となっている．この熱揺らぎの問題を解決する方法としてビットパターンドメディア (BPM) が開発されている [1]．BPM は磁性粒子 (ビットアイランド) を規則的に配置し，隣接する粒子を非磁性領域 (トレンチ) で分断する記憶媒体であり，データ 1 ビットがひとつのビットアイランドに書き込まれる．ビットパターンドメディアレコーディング (BPMR) では，書き込むデータのタイミングと回転するディスク上のビットアイランドのタイミングを同期させる必要がある．内部クロックのゆらぎ，磁気ヘッドの振動，ディスクの回転速度の変化やビットアイランドの位置や大きさのばらつきは同期ミスの原因となり，記録されたデータに削除／挿入誤りが生じる場合がある [2]．

記録誤りに対する有効な手段として，誤り訂正符号の適用が挙げられる．BPMR では，削除／挿入誤りのほかに，反転誤りも考慮する必要があり，十分な削除／挿入／反転誤り訂正能力が求められる．しかし，従来のハードディスクに適用されていたファイア符号や RS 符号など，既存の符号の多くは削除／挿入誤りを訂正できない．BPMR に対する誤り訂正符号として，Picket-shift 符号 [3] が提案されているが，削除と挿入の同時誤りについては考慮されていない．

本稿では，単一削除／挿入／反転誤り訂正符号である Levenshtein 符号 [4],[5] をもとに，二重削除／挿入／反転誤り訂正符号を提案し，その符号化法と復号法を示す．また，シミュレーションにより復号誤り率を評価する．

本稿の構成は以下のとおりである．**2** で BPMR のモデルと削除／挿入／反転誤りについて述べる．**3** では Levenshtein 符号について述べ，二重削除／挿入／反転誤り訂正符号とその復号法を提案する．**4** で評価結果を示し，**5** で結論と今後の課題を明らかにする．

2 準備

本節では，BPMR のモデルを示し，その上での削除／挿入／反転誤りについて述べる．

2.1 BPMR のモデル化

図 1 に BPM を用いたハードディスクの構成と半径 r のトラック上のビットアイランドの配置を示す．ここで，ビットアイランドはビット値を保持する磁性領域であり，トレンチは隣接するビットアイランドを分断する非磁性領域である．ビットアイランド及びトレンチのトラック方向の長さをそれぞれ w^b 及び w^t とする．ドライブのヘッドが半径 r のトラック上にあるとき，ヘッドとトラック上

*東京工業大学 大学院情報理工学研究科

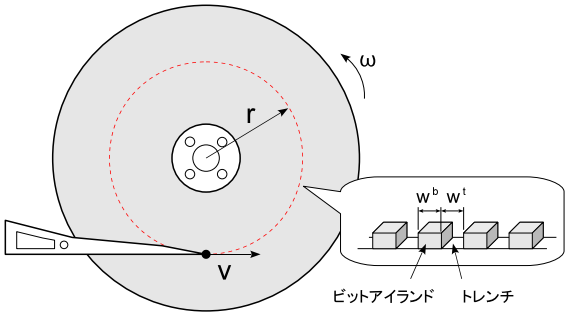

図 1 BPM を用いたハードディスクの構成

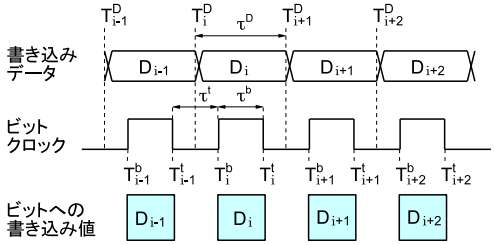

図 2 理想的な書き込み

のビットアイランドの相対速度は $v = r\omega$ となる．よって，ヘッドの直下をビットアイランド及びトレンチが通過する時間はそれぞれ $\tau^b = w^b/v$ 及び $\tau^t = w^t/v$ となり，書き込みデータの周期は $\tau^D = \tau^b + \tau^t$ となる．書き込みデータの第 j ビットの値を $D_j \in \{0,1\}$ とし，その書き込み開始時刻を T_j^D とすると，数列 $\{\cdots, T_{j-1}^D, T_j^D, T_{j+1}^D, \cdots\}$ は以下の漸化式で与えられる．

$$T_j^D = T_{j-1}^D + \tau^D$$

また，第 i 番目のビットアイランドとそれに続くトレンチの各領域の開始位置がヘッドの直下に到達する時刻をそれぞれ T_i^b 及び T_i^t とすると，数列 $\{\cdots, T_{i-1}^b, T_i^b, T_{i+1}^b, \cdots\}$ 及び数列 $\{\cdots, T_{i-1}^t, T_i^t, T_{i+1}^t, \cdots\}$ はそれぞれ以下の漸化式で与えられる．

$$T_i^b = T_{i-1}^t + \tau^t$$
$$T_i^t = T_i^b + \tau^b$$

理想的には，書き込みデータのエッジがトレンチの中央にあるとする．すなわち

$$T_i^D = T_{i-1}^t + \tau^t/2 \qquad (1)$$

である．時刻 T_i^D,T_i^b 及び T_i^t の関係を図 2 に示す．ここで，ビットクロックとはヘッドの直下にビットアイランドがあるかトレンチがあるかを時刻 t の関数として示したものであり，次式で定義する．

$$b(t) = \begin{cases} \text{High} & (\exists i \;\; T_i^b \le t < T_i^t) \\ \text{Low} & (\text{otherwise}) \end{cases}$$

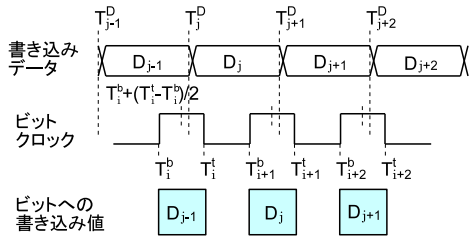

図 3　書き込み値が式 (3) の第一式で与えられる場合の例

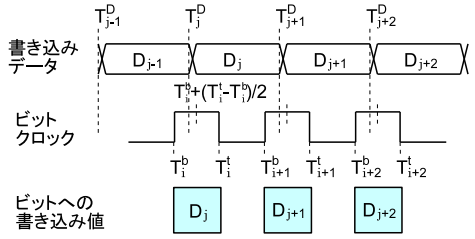

図 4　書き込み値が式 (3) の第二式で与えられる場合の例

上記で定めた時刻 T_j^D，T_i^b 及び T_i^t は誤差を考慮しない理想的なモデルである．実際のシステムでは内部クロックの誤差によりデータの周期に誤差が生じる．さらに，磁気ヘッドの振動，ディスクの回転速度の変化，ビットアイランドの位置や大きさのばらつきにより，ビットクロックにも誤差が生じる．これらの誤差を考慮し，時刻 T_j^D，T_i^b 及び T_i^t を次の漸化式で定義する．

$$\begin{aligned} T_j^D &= T_{j-1}^D + \rho^D \\ T_i^b &= T_{i-1}^t + \rho^t \\ T_i^t &= T_i^b + \rho^b \end{aligned} \tag{2}$$

ここで，ρ^D，ρ^b 及び ρ^t は，正規分布に従う確率変数であり，それぞれの平均は τ^D,τ^b 及び τ^t ，分散は ${\sigma^D}^2$,${\sigma^b}^2$ 及び ${\sigma^t}^2$ である．

ビットアイランドへ書き込まれる値 V_i は，そのビットアイランドの 50% 以上が含まれるデータ周期が持つデータの値とする．ビットアイランドの 50% 以上に相当するデータ周期が存在しない場合は，0 または 1 がランダムに書き込まれる．すなわち

$$V_i = \begin{cases} D_{j-1} & \left(T_{j-1}^D \le T_i^b \text{ かつ } T_i^b + \frac{T_i^t - T_i^b}{2} < T_j^D\right) \\ D_j & \left(T_j^D \le T_i^b + \frac{T_i^t - T_i^b}{2} \text{ かつ } T_i^t < T_{j+1}^D\right) \\ 0 \text{ or } 1 \text{ (otherwise)} \end{cases} \tag{3}$$

である．この例を図 3 及び図 4 に示す．

2.2　BPMR における削除／挿入／反転誤り

図 5 及び図 6 に削除／挿入誤りの例を示す．図 5 に示すように，書き込みデータの周期に対して，ビットクロックの周期が長い場合は削除誤りが生じ，例ではデータ D4 が欠落している．一方，図 6 に示すように，ビットクロックの周期が短い場合は挿入誤りが生じ，例ではデータ D4 が 2 回書き込まれている．

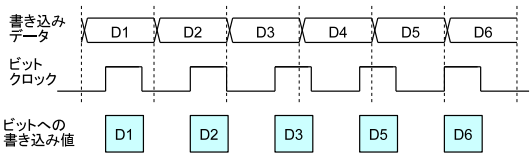

図 5　削除誤り

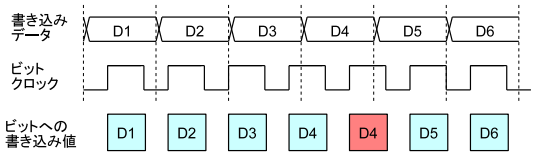

図 6　挿入誤り

上記の削除／挿入誤りの他に，各ビットの値が反転する反転誤りを考える．この誤りは，ビットアイランドの磁化方向を反転させるのに不十分なヘッド磁界や，ヘッドからの漏れ磁界による隣接ビットアイランドの磁化反転などが原因として挙げられる [6]．したがって，ビットアイランドに書き込まれたデータは，ある確率 P_{bsc} でその値が反転する．

3　削除／挿入／反転誤り訂正符号

以下では，従来の誤り訂正符号である Levenshtein の単一削除／挿入／反転誤り訂正符号と，提案する二重削除／挿入／反転誤り訂正符号について述べる．

3.1　誤りの定義

削除／挿入／反転誤り，及び削除誤りと挿入誤りの同時誤りである左シフト／右シフト誤りを定義する．

以下では符号長 n の符号語

$$\boldsymbol{c} = c_1\ c_2\ \cdots\ c_{i-1}\ c_i\ c_{i+1}\ \cdots\ c_n \tag{4}$$

における誤りを考える．ここで，$c_i \in \{0,1\}$ である．

削除誤り

削除誤りとは，符号語中のビットが削除され受信語長が短くなる誤りである．符号語を式 (4) で表したとき，c_i における削除誤りとは，符号語中の c_i が削除され，c_{i+1} 以降が左に 1 ビットずれるような誤りである．このとき受信語は

$$\boldsymbol{r} = c_1\ c_2\ \cdots\ c_{i-1}\ c_{i+1}\ \cdots\ c_n$$

となる．

挿入誤り

挿入誤りとは，符号語中に余分なビットが挿入され受信語長が長くなる誤りである．符号語を式 (4) で表したとき，c_i における挿入誤りとは，符号語中の c_i にビット $I \in \{0,1\}$ が挿入され，c_i 以降が右に 1 ビットずれるような誤りである．このとき受信語は

$$\boldsymbol{r} = c_1\ c_2\ \cdots\ c_{i-1}\ I\ c_i\ c_{i+1}\ \cdots\ c_n$$

となる．また，符号語の最後に挿入誤りが発生した場合の受信語は，

$$\boldsymbol{r} = c_1\ c_2\ \cdots\ c_{i-1}\ c_i\ c_{i+1}\ \cdots\ c_n\ I$$

となる．

反転誤り

反転誤りとは，符号語中のビットが反転する誤りである．符号語を式 (4) で表したとき，c_i における反転誤りとは，c_i の値が $\overline{c_i}$ となるような誤りである．ただし，$\overline{c_i}$ は c_i の反転である．このとき受信語は，

$$\boldsymbol{r} = c_1\ c_2\ \cdots\ c_{i-1}\ \overline{c_i}\ c_{i+1}\ \cdots\ c_n$$

となる．

左シフト誤り

左シフト誤りとは，符号語の一部が左シフトした形になる誤りである．すなわち，符号語を式 (4) で表したとき，c_i における削除誤りと c_j における挿入誤りが同時に存在するような誤りである．ただし，$i < j$ である．このとき受信語は，

$$\boldsymbol{r} = c_1\ c_2\ \cdots\ c_{i-1}\ c_{i+1}\ \cdots\ c_{j-1}\ I\ c_j\ c_{j+1}\ \cdots\ c_n$$

となる．

右シフト誤り

右シフト誤りとは，符号語の一部が右シフトした形になる誤りである．すなわち，符号語を式 (4) で表したとき，c_i における挿入誤りと c_j における削除誤りが同時に存在するような誤りである．ただし，$i < j$ である．このとき受信語は，

$$\boldsymbol{r} = c_1\ c_2\ \cdots\ c_{i-1}\ I\ c_i\ c_{i+1}\ \cdots\ c_{j-1}\ c_{j+1}\ \cdots\ c_n$$

となる．

3.2 Levenshtein の単一削除／挿入／反転誤り訂正符号

Levenshtein 符号は単一削除／挿入／反転誤りを訂正可能な二元組織符号である [4],[5]．本稿では，この符号をもとに二重削除／挿入／反転誤り訂正符号を提案する．

Levenshtein 符号 [4] は，次式を満たすような符号語 $\boldsymbol{c} = c_1c_2\cdots c_n$ の集合として定義される．

$$\sum_{i=1}^{n} i \cdot c_i \equiv \alpha \pmod{M} \tag{5}$$

ここで，$c_i \in \{0,1\}$ であり，α 及び M は整数であり，$0 \le \alpha \le M-1, M \ge 2n$ である．以降では，$\alpha = 0$ とする．

単一削除／挿入／反転誤り訂正 Levenshtein 符号の符号化と復号法 [5] を以下に示す．

3.2.1 符号化アルゴリズム

符号語の構成を図 7 に示す．情報語を $d_1d_2\cdots d_k$，検査ビットを $p_1p_2\cdots p_r$，符号語を $c_1c_2\cdots c_n$ $(n = k+r)$ とすると，符号語の各ビット c_i は以下のように定義される．

$$c_i = \begin{cases} d_{i-\lceil \log_2 i \rceil} & \text{for } i \neq n \text{ and } i \neq 2^0, 2^1, \ldots, 2^{r-2} \\ p_{\log_2 i + 1} & \text{for } i = 2^0, 2^1, \ldots, 2^{r-2} \\ p_r & \text{for } i = n \end{cases}$$

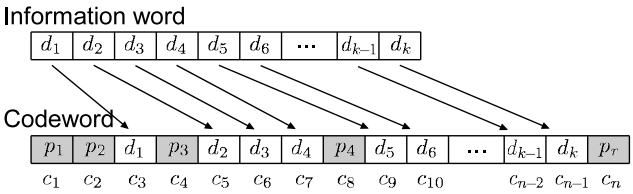

図 7　Levenshtein 符号の符号語の構成

ただし，実数 x に対して $\lceil x \rceil$ は x 以上の最小の整数を表し，r は $k + r \le 2^{r-1}$ を満たす最小の整数である．検査ビットは符号語が式 (5) を満たすように，以下のように決定される．

$$\sum_{j=0}^{r-2} p_{j+1} \cdot 2^j + p_r \cdot n + \sum_{\substack{i=1 \\ i \neq 2^0, 2^1, \ldots, 2^{r-2}}}^{n-1} i \cdot d_{i-\lceil \log_2 i \rceil} \equiv 0 \pmod{M}$$

3.2.2 復号アルゴリズム

受信語を $r_1r_2\cdots r_{n'}$，受信語のハミング重みを W，シンドロームを $S = \left(\sum_{i=1}^{n'} i \cdot r_i\right) \bmod M$ とする．受信語の長さより誤りの種類を以下のように決定する．

(1) $n' = n-1 \longrightarrow$ 単一削除誤り
(2) $n' = n+1 \longrightarrow$ 単一挿入誤り
(3) $n' = n \quad\ \longrightarrow$ 誤りなし，または単一反転誤り

ここで，受信語の長さはマーカーなどにより既知であると仮定している．各誤りに対して以下のとおり訂正を行う．

(1) $n' = n-1$ の場合：

削除されたビットの値 z を以下のように決定する．

$$z = \begin{cases} 0 & (\ (-S) \bmod M \le W) \\ 1 & (\ (-S) \bmod M > W) \end{cases}$$

$X_{\mathrm{del}}(t)$ を $r_t r_{t+1}\cdots r_{n'}$ のハミング重みとし，

$$(-S) \bmod M = t \cdot z + X_{\mathrm{del}}(t)$$

を満たす t を求め，受信語の第 $t-1$ ビットと第 t ビットの間に z を挿入した語を復号語とする．

(2) $n' = n+1$ の場合：

挿入されたビットの値 z を以下のように決定する．

$$z = \begin{cases} 0 & (S < W) \\ 1 & (S > W) \\ r_1 & (S = W) \end{cases}$$

$X_{\mathrm{ins}}(t)$ を $r_{t+1}r_{t+2}\cdots r_{n'}$ のハミング重みとし，

$$S = t \cdot z + X_{\mathrm{ins}}(t)$$

を満たす t を求める．$r_t = z$ であるならば，受信語の第 t ビットを削除した語を復号語とする．

(3) $n' = n$ の場合：

$S = 0$ ならば，受信語に誤りは存在しないので，受信語が復号語となる．

$0 < S \leq n$ かつ $r_S = 1$ ならば，受信語の第 S ビットの値を反転した語を復号語とする．

$M - n \leq S < M$ かつ $r_{M-S} = 0$ ならば，受信語の第 $M - S$ ビットの値を反転した語を復号語とする．

3.3 提案する符号の定義

符号長 n の Levenshtein 符号を次式で定義する．

$$\boldsymbol{C'} = \{\boldsymbol{c'} = c'_1 c'_2 \cdots c'_n \mid \Sigma_{i=1}^{n} i \cdot c'_i \equiv 0 \pmod{M}\} \tag{6}$$

ただし，$M \geq 2n$ である．$\boldsymbol{C'}$ の符号語の各ビットを 2 回繰り返して得られる符号長 $2n$ の符号 $\boldsymbol{C}$ を次式で定義する．

$$\boldsymbol{C} = \{\, \boldsymbol{c} = a_1 b_1 a_2 b_2 \cdots a_n b_n \mid a_i = b_i = c'_i \ (1 \leq i \leq n),\ c'_1 c'_2 \cdots c'_n \in \boldsymbol{C'} \,\} \tag{7}$$

3.4 提案する符号の復号法

本節では，式 (7) に定義する符号に対する二重削除／挿入／反転誤り訂正アルゴリズムを示す．

3.4.1 準備

以降で用いる表記を定義する．符号語を

$$\boldsymbol{c} = a_1\ b_1\ a_2\ b_2\ \cdots\ a_{i-1}\ b_{i-1}\ a_i\ b_i\ a_{i+1}\ b_{i+1}\ \cdots\ a_n\ b_n \tag{8}$$

と表記する．ただし，$a_i = b_i = c'_i,\ c'_1 c'_2 \cdots c'_n \in \boldsymbol{C'}$ である．受信語 $\boldsymbol{r} = r_1 r_2 \cdots r_{n'}$ の奇数番目のビットからなるベクトルを

$$A(\boldsymbol{r}) = \boldsymbol{r^A} = r_1^A\ r_2^A\ \cdots\ r_i^A\ \cdots = r_1\ r_3\ \cdots\ r_{2i-1}\ \cdots$$

と表記し，偶数番目からなるベクトルを

$$B(\boldsymbol{r}) = \boldsymbol{r^B} = r_1^B\ r_2^B\ \cdots\ r_i^B\ \cdots = r_2\ r_4\ \cdots\ r_{2i}\ \cdots$$

と表記する．例えば，符号語 $\boldsymbol{c}$ の a_i にビット $I \in \{0,1\}$ が挿入された場合，ベクトル $\boldsymbol{r^A}, \boldsymbol{r^B}$ は以下のようになる．

$$\begin{aligned} \boldsymbol{r^A} &= a_1 a_2 \cdots a_{i-1}\ I\ \ b_i\ \ \cdots b_{n-1} b_n = c'_1 c'_2 \cdots c'_{i-1}\ I\ \ c'_i\ \ \cdots c'_{n-1} c'_n \\ \boldsymbol{r^B} &= b_1 b_2 \cdots b_{i-1} a_i a_{i+1} \cdots\ a_n\ \ \ \ = c'_1 c'_2 \cdots c'_{i-1} c'_i c'_{i+1} \cdots\ c'_n \end{aligned}$$

このとき $\boldsymbol{r^A}$ は Levenshtein 符号の符号語に単一挿入誤りが生じたものとなり，$\boldsymbol{r^B}$ は Levenshtein 符号の符号語となる．

受信語の奇数ビット目から 3 ビットずつ取り出したビットの組 $r_{2m-1} r_{2m} r_{2m+1}$ をフレームとし，このフレームを並べたものを

$$r_1 r_2 r_3,\ r_3 r_4 r_5,\ \cdots,\ r_{2i-3} r_{2i-2} r_{2i-1},\ r_{2i-1} r_{2i} r_{2i+1},\ r_{2i+1} r_{2i+2} r_{2i+3},\ \cdots$$

とする．ただし，フレームは 1 ビットずつオーバーラップする．各フレームにおいて多数決をとって構成したベクトルを

$$X(\boldsymbol{r}) = \boldsymbol{x} = x_1\ x_2\ \cdots\ x_{i-1}\ x_i\ x_{i+1}\ \cdots$$

と表記する．ただし，x_m は以下のように定義する．

$$x_m = \mathrm{majority}(r_{2m-1}, r_{2m}, r_{2m+1}) \tag{9}$$

ここで，$\mathrm{majority}(x, y, z)$ は $x, y, z \in \{0,1\}$ の多数決をとる関数である．また，ベクトル $\boldsymbol{r^A}$,$\boldsymbol{r^B}$,$\boldsymbol{x}$ の長さが n である場合，シンドローム S_A, S_B, S_x を以下のように定義する．

$$S_A = \left(\sum_{i=1}^{n} i \cdot r_i^A\right) \bmod M$$

$$S_B = \left(\sum_{i=1}^{n} i \cdot r_i^B\right) \bmod M$$

$$S_x = \left(\sum_{i=1}^{n} i \cdot x_i\right) \bmod M$$

単一削除／挿入／反転誤りを有するベクトル $\boldsymbol{r^A}$,$\boldsymbol{r^B}$,$\boldsymbol{x}$ を Levenshtein 符号の復号手順で復号した語を $\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$,$\tilde{\boldsymbol{r}}^{\boldsymbol{B}}$,$\tilde{\boldsymbol{x}}$ と表記する

3.4.2 復号アルゴリズム

以下では復号器において受信語長が既知であるとする．すなわち，マーカー等により受信語の開始位置と終了位置がわかるとする．以下では誤り訂正の方法について，受信語の長さで場合分けして述べる．符号長を $2n$，受信語長を n' とすると，n' の値で場合分けしたときに，以下のような誤りの組合せを考えることができる．

(1) $n' = 2n - 1$ ：単一削除誤り，単一削除と単一反転の同時誤り

(2) $n' = 2n + 1$ ：単一挿入誤り，単一挿入と単一反転の同時誤り

(3) $n' = 2n - 2$ ：二重削除誤り

(4) $n' = 2n + 2$ ：二重挿入誤り

(5) $n' = 2n$ ：誤りなし，2 個以下の反転誤り，単一削除と単一挿入の同時誤り

上記の各場合に対して，受信語 $\boldsymbol{r} = r_1 r_2 \cdots r_{n'}$ について，以下のように復号を行う．

(1) $\boldsymbol{n' = 2n - 1}$ の場合

$A(\boldsymbol{r}) = \boldsymbol{r^A} = r_1^A r_2^A \cdots r_n^A$ のシンドローム S_A を計算する．$S_A = 0$ の場合は $\boldsymbol{r^A}$ を復号語とする．そうでない場合は，Levenshtein 符号の復号手順により $\boldsymbol{r^A}$ における反転誤りを訂正し，$\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$ を復号語とする．

(2) $\boldsymbol{n' = 2n + 1}$ の場合

$X(\boldsymbol{r}) = \boldsymbol{x} = x_1 x_2 \cdots x_n$ のシンドローム S_x を計算する．$S_x = 0$ の場合は $\boldsymbol{x}$ を復号語とする．そうでない場合は，Levenshtein 符号の復号手順により $\boldsymbol{x}$ における反転誤りを訂正し，$\tilde{\boldsymbol{x}}$ を復号語とする．

(3) $\boldsymbol{n' = 2n - 2}$ の場合

Levenshtein 符号の復号手順で $A(\boldsymbol{r}) = \boldsymbol{r^A} = r_1^A r_2^A \cdots r_{n-1}^A$ における削除誤りを訂正し，$\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$ を復号語とする．

(4) $\boldsymbol{n' = 2n + 2}$ の場合

$A(\boldsymbol{r}) = \boldsymbol{r^A} = r_1^A r_2^A \cdots r_{n+1}^A$，$B(\boldsymbol{r}) = \boldsymbol{r^B} = r_1^B r_2^B \cdots r_{n+1}^B$，$X(\boldsymbol{r}) = \boldsymbol{x} = x_1 x_2 \cdots x_{n+1}$ とする．ベクトル $\boldsymbol{r'} = r'_1 r'_2 \cdots r'_{n+1}$ を以下のように決定する．

$$r'_m = \begin{cases} r_m^B & (x_{m-1} \neq r_{m-1}^A \text{ または } x_{m-1} \neq r_{m-1}^B) \\ x_m & (\text{otherwise}) \end{cases} \quad (10)$$

Levenshtein 符号の復号手順で $\boldsymbol{r'}$ における挿入誤りを訂正した語 $\tilde{\boldsymbol{r}}'$ を復号語とする．

(5) $\boldsymbol{n' = 2n}$ の場合

$A(\boldsymbol{r}) = \boldsymbol{r^A} = r_1^A r_2^A \cdots r_n^A$，$B(\boldsymbol{r}) = \boldsymbol{r^B} = r_1^B r_2^B \cdots r_n^B$ のシンドローム S_A, S_B を計算する．

(I) $S_A = 0$ または $S_B = 0$ である場合 ：

(I.a) $S_A = 0$ ならば $\boldsymbol{r^A}$ を復号語とする．

(I.b) $S_B = 0$ ならば $\boldsymbol{r^B}$ を復号語とする．

(Ⅱ) $S_A \neq 0$ かつ $S_B \neq 0$ である場合 ：
$\boldsymbol{r_A}$,$\boldsymbol{r_B}$ における反転誤りの訂正を行う．

(Ⅱ.a) $\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$=$\tilde{\boldsymbol{r}}^{\boldsymbol{B}}$ ならば，$\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$=$\tilde{\boldsymbol{r}}^{\boldsymbol{B}}$ を復号語とする．

(Ⅱ.b) $\tilde{\boldsymbol{r}}^{\boldsymbol{A}} \neq \tilde{\boldsymbol{r}}^{\boldsymbol{B}}$ ならば，受信語においてベクトル $X(\boldsymbol{r}) = \boldsymbol{x} = x_1 x_2 \cdots x_n$ を求める．
次に，$\boldsymbol{r^B}$ と $\boldsymbol{x}$ のハミング距離 $d(\boldsymbol{r^B}, \boldsymbol{x})$ を計算し，以下のとおり復号を行う．

- $d(\boldsymbol{r^B}, \boldsymbol{x}) = 2$ のとき，
  - $\boldsymbol{r^B}$ と $\boldsymbol{x}$ で異なるビットが隣接していないならば，$\boldsymbol{r^B}$ の反転誤りを訂正し，$\tilde{\boldsymbol{r}}^{\boldsymbol{B}}$ を復号語とする．
  - $\boldsymbol{r^B}$ と $\boldsymbol{x}$ で異なるビットが隣接しているならば，$\boldsymbol{r^A}$ の反転誤りを訂正し，$\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$ を復号語とする．
- $d(\boldsymbol{r^B}, \boldsymbol{x}) = 1$ のとき，$\boldsymbol{x}$ のシンドローム S_x を計算する
  - $S_x = 0$ ならば，$\boldsymbol{x}$ を復号語とする．
  - $S_x \neq 0$ ならば，$\boldsymbol{r^A}$ の反転誤りを訂正し，$\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$ を復号語とする．
- $d(\boldsymbol{r^B}, \boldsymbol{x}) = 0$ ならば，$\boldsymbol{r^A}$ の反転誤りを訂正し，$\tilde{\boldsymbol{r}}^{\boldsymbol{A}}$ を復号語とする．

3.4.3 復号アルゴリズムの正当性についての略証

3.4.2 に示した復号アルゴリズムにより，二重削除／挿入／反転誤りが訂正可能であることの略証を以下に示す．

(1) $n' = 2n - 1$ の場合：

このとき受信語には単一削除誤りが存在し，符号語のある位置から右側が 1 ビット左にずれた状態となる．符号語 $\boldsymbol{c}$ は Levenshtein 符号の符号語 $\boldsymbol{c'}$ の各ビットを 2 回繰り返したものであるから，ある位置から右側が 1 ビット左にずれたとしても，受信語 $\boldsymbol{r}$ の奇数番目のビットからなるベクトル $A(\boldsymbol{r}) = \boldsymbol{r^A}$ は変化せず，$\boldsymbol{r^A}$=$\boldsymbol{c'}$ となる．削除誤りに加えて反転誤りが生じた場合でも，$\boldsymbol{r^A}$ は $\boldsymbol{c'}$ と比較して高々1 ビットの反転誤りにしかならない．したがって，Levenshtein 符号の復号手順により訂正可能である．

(2) $n' = 2n + 1$ の場合：

式 (8) における符号語のフレームを並べたものは，

$$a_1 b_1 a_2,\ a_2 b_2 a_3,\ \cdots,\ a_{i-1} b_{i-1} a_i,\ a_i b_i a_{i+1},\ a_{i+1} b_{i+1} a_{i+2},\ \cdots,\ a_n b_n \quad (11)$$

となる．$a_k = b_k = c'_k$ $(k = 1, 2, \ldots, n)$ であるから，それぞれのフレームで多数決をとって並べると，

$$c'_1\ c'_2\ \cdots\ c'_{i-1}\ c'_i\ c'_{i+1}\ \cdots\ c'_n$$

となり，元の Levenshtein 符号の符号語 $\boldsymbol{c'}$ と一致する．

$n' = 2n + 1$ となるとき，受信語には単一挿入誤りが存在し，符号語がある位置から右側が 1 ビット右にずれた状態となる．式 (11) の各フレームにおいて，$a_k = b_k$ の要素はフレームの左に寄っている．よって，挿入誤りの位置より右側のフレームにおいても，多数決をとった結果は，符号語のフレームで多数決をとった結果と一致する．挿入誤りに加えて反転誤りが生じた場合でも，各フレームにおいて多数決を取った結果は $\boldsymbol{c'}$ と比較して高々1 ビットの反転誤りにしかならない．したがって，Levenshtein 符号の復号手順により訂正可能である．

(3) $n' = 2n - 2$ の場合：

このとき受信語には二重削除誤りが存在し，最初の削除誤りによって，符号語はこの位置から右側が 1 ビット左にずれ，次の削除誤りによって，その位置から右側がさらに 1 ビット左にずれた状態となる．受信語 $\boldsymbol{r}$ の奇数ビット目からなるベクトル $A(\boldsymbol{r}) = \boldsymbol{r^A}$ において，先頭から 2 つ目の削除誤り位置までの $\boldsymbol{r^A}$ の要素は，元の Levenshtein 符号の符号語 $\boldsymbol{c'}$ の要素と一致する．2 つ目の削除誤り位置より右側では，受信語は符号語と比較して 2 ビット左にずれるので，$\boldsymbol{r^A}$ の残りの要素は，$\boldsymbol{c'}$ の残りの要素と比較して，1 ビット左にずれる．したがって，$\boldsymbol{r^A}$ は $\boldsymbol{c'}$ に単一削除誤りが発生したものになるので，Levenshtein 符号の復号手順により訂正可能である．

(4) $n' = 2n + 2$ の場合：

式 (10) で定まる $\boldsymbol{r'} = r'_1 r'_2 \cdots r'_{n+1}$ が単一挿入誤りを有することを示す．

符号語の第 i ビットと第 j ビット $(i \leq j)$ で挿入誤りが発生したとする．このとき，r_m^B, x_m $(m = 1, 2, \ldots, n+1)$

の値は次式で与えられる．

$$r^B_m = \begin{cases} c'_m & (m \le \lceil i/2 \rceil - 1) \\ R^B_{\lceil i/2 \rceil} & (m = \lceil i/2 \rceil) \\ c'_m & (\lceil i/2 \rceil + 1 \le m \le \lceil j/2 \rceil - 1) \\ R^B_{\lceil j/2 \rceil} & (m = \lceil j/2 \rceil) \\ c'_{m-1} & (\lceil j/2 \rceil + 1 \le m) \end{cases}$$

$$x_m = \begin{cases} c'_m & (m \le \lceil j/2 \rceil - 1) \\ X_{\lceil j/2 \rceil} & (m = \lceil j/2 \rceil) \\ X_{\lceil j/2 \rceil + 1} & (m = \lceil j/2 \rceil + 1) \\ c'_{m-1} & (\lceil j/2 \rceil + 2 \le m) \end{cases}$$

ただし，$R^B_{\lceil i/2 \rceil}, R^B_{\lceil j/2 \rceil}, X_{\lceil j/2 \rceil}, X_{\lceil j/2 \rceil + 1} \in \{0, 1\}$ は，i, j 及び挿入されるビットの値で決まる．

以上より，$\boldsymbol{r^B}$ と $\boldsymbol{x}$ では第 $\lceil i/2 \rceil$ ビット，第 $\lceil j/2 \rceil$ ビット，第 $\lceil j/2 \rceil + 1$ ビット以外は一致するため，$\boldsymbol{r'}$ は $\boldsymbol{x}$ と第 $\lceil j/2 \rceil$ ビット，第 $\lceil j/2 \rceil + 1$ ビット以外一致する．したがって，

$$r'_m = \begin{cases} x_m = c'_m & (m < \lceil j/2 \rceil) \\ R_{\lceil j/2 \rceil} & (m = \lceil j/2 \rceil) \\ R_{\lceil j/2 \rceil + 1} & (m = \lceil j/2 \rceil + 1) \\ x_m = c'_{m-1} & (\lceil j/2 \rceil + 1 < m) \end{cases}$$

となる．よって，$R_{\lceil j/2 \rceil} = c'_{\lceil j/2 \rceil}$ または $R_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ が成り立つことを示す．

(i) $X_{\lceil j/2 \rceil} = c'_{\lceil j/2 \rceil}$ の場合：
$x_{\lceil j/2 \rceil} = r^B_{\lceil j/2 \rceil}$ ならば，$R_{\lceil j/2 \rceil} = c'_{\lceil j/2 \rceil}$ となる．一方，$x_{\lceil j/2 \rceil} \neq r^B_{\lceil j/2 \rceil}$ ならば，$R_{\lceil j/2 \rceil + 1} = r^B_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ となる．

(ii) $X_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ の場合：
$x_{\lceil j/2 \rceil + 1} = r^B_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ が成り立つため，$r'_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ となる．

(iii) $X_{\lceil j/2 \rceil} = X_{\lceil j/2 \rceil + 1} \neq c'_{\lceil j/2 \rceil}$ の場合：
$x_{\lceil j/2 \rceil} \neq r^B_{\lceil j/2 \rceil}$ ならば，$R_{\lceil j/2 \rceil + 1} = r^B_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ となる．一方，i が偶数で $j = i$ の場合，$x_{\lceil j/2 \rceil} = r^B_{\lceil j/2 \rceil}$ となりうる．この場合，$r^A_{\lceil j/2 \rceil} = c'_{\lceil j/2 \rceil}$ が成立することが確かめられる．よって，$x_{\lceil j/2 \rceil} \neq r^A_{\lceil j/2 \rceil}$ より，$R_{\lceil j/2 \rceil + 1} = r^B_{\lceil j/2 \rceil + 1} = c'_{\lceil j/2 \rceil}$ となる．

したがって，$\boldsymbol{r'}$ は単一挿入誤りを有するので，Levenshtein 符号の復号手順で訂正可能である．

(5) $n' = 2n$ の場合：

このとき，$\boldsymbol{c'}$ と比較して，$\boldsymbol{r^A}$,$\boldsymbol{r^B}$ のどちらか一方には，高々一個の反転誤りしか存在しないことを示す．

(i) 受信語に単一反転誤りが存在する場合：
$\boldsymbol{r^A}$,$\boldsymbol{r^B}$ のどちらかに単一反転誤りが存在し，もう一方には誤りは存在しない．

(ii) 受信語に二重反転誤りが存在する場合：
反転したビットが両方とも偶数ビット目あるいは奇数ビット目ならば，$\boldsymbol{r^A}$,$\boldsymbol{r^B}$ のどちらかに二重反転誤りが存在し，もう一方には誤りは存在しない．一方，反転したビットが偶数ビット目と奇数ビット目ならば，$\boldsymbol{r^A}$,$\boldsymbol{r^B}$ に単一反転誤りが存在する．

(iii) 受信語に削除誤りと挿入誤りが存在する場合：
第 i ビットに削除誤り，第 j ビットに挿入誤りが存在するとする．

(iii.a) $i < j$ の場合：
受信語は左シフト誤りを有する．このとき $\boldsymbol{r^A}$ に注目すると，受信語の左シフトした部分で抽出する値は，$\boldsymbol{r^A}$ が本来とるべき値と一致する．ただし，j が偶数かつ挿入されたビットの値が $I \neq c_j$ の場合，$\boldsymbol{r^A}$ は第 $j/2$ ビットに単一反転誤りを有する．よって，$\boldsymbol{r^A}$ には高々1ビットの反転誤りしか存在しない．

(iii.b) $j < i$ の場合：
受信語は右シフト誤りを有する．このとき $\boldsymbol{r^B}$ に注目すると，受信語の右シフトした部分で抽出する値は，$\boldsymbol{r^B}$ が本来とるべき値と一致する．ただし，j が偶数かつ挿入されたビットの値が $I \neq c_j$ の場合，$\boldsymbol{r^B}$ は第 $j/2$ ビットに単一反転誤りを有する．よって，$\boldsymbol{r^B}$ には高々1ビットの反転誤りしか存在しない．

以上より，$\boldsymbol{r^A}$,$\boldsymbol{r^B}$ のどちらか一方には，高々一個の反転誤りしか存在しない．

復号アルゴリズムはこれに基づいて，$\boldsymbol{r^A}$,$\boldsymbol{r^B}$ のどちらが単一反転誤りを有するかを判別し，Levenshtein 符号の復号手順により反転誤りを訂正する．

3.5 制御能力外誤りに対する訂正能力

提案した符号は前述の復号アルゴリズムにより，二重削除／挿入／反転誤りを訂正可能であるが，制御能力外の誤りも一部訂正可能である．受信語長を n' とし，以下に訂正可能な誤りと訂正方法を示す．

(i) $n' = 2n$ の場合：

- 誤りのパターン：$\boldsymbol{r^A}$ に誤りなし，$\boldsymbol{r^B}$ に多ビット反転誤り
  訂正方法：$\boldsymbol{r^A}$ を復号語とする
- 誤りのパターン：$\boldsymbol{r^B}$ に誤りなし，$\boldsymbol{r^A}$ に多ビット反転誤り
  訂正方法：$\boldsymbol{r^B}$ を復号語とする

(ii) $n' = 2n - 3$ の場合：

- 誤りのパターン：三重削除誤り
  訂正方法：$\boldsymbol{r^A}$ における削除誤りを訂正

(iii) $n' = 2n + 3$ の場合：

- 誤りのパターン：三重挿入誤り（ただし挿入ビットの値は挿入位置のビットの値）
  訂正方法：$\boldsymbol{r^B}$ における挿入誤りを訂正

4 復号誤り率の評価

Levenshtein 符号と提案する符号を用いた場合について，BPMR のシミュレーションを行い，誤り訂正を行った際の復号誤り率を評価する．

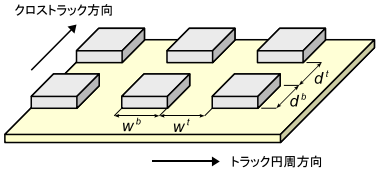

図 8　Levenshtein 符号を用いる場合の配置

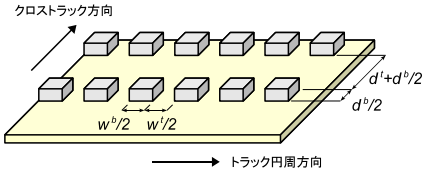

図 9　提案する符号を用いる場合の配置

4.1　シミュレーションの条件

以下のシミュレーションでは，1 ブロック当たり 4096 バイトの情報を持つハードディスクを対象にする．また，1 ブロックを 24 分割し，情報語 171 バイト (1368 ビット) に対し，符号化を行い，書き込みデータとする．したがって，Levenshtein 符号で符号化する場合は，符号長は 1380 ビットであり，提案する符号で符号化する場合は，符号長は 2760 ビットである．サーボパターンによって，書き込みデータの先頭ではデータのエッジが式 (1) に示す理想的な状態に合わせられると仮定し，トラック円周方向のビットアイランドへ記録される．データの再生は誤りなく行われるとし，ビットアイランドへ記録されたデータが受信語となる．ここで，サーボパターンやマーカーによって，受信語の長さを判別できると仮定する．

図 8 及び図 9 に Levenshtein 符号及び提案する符号を用いる場合のディスク上のビットアイランドの配置を示す．提案する符号の符号長は，Levenshtein 符号の符号長と比べて 2 倍になる．したがって，情報語の記憶密度を同一にするために，提案する符号を用いる場合は，Levenshtein 符号を用いる場合と比べて，ビットアイランドのサイズを縦横それぞれ半分とする．

式 (2) 及び式 (3) で表されるクロック及び書き込みモデルを用いて，書き込みを行う．ここで，式 (2) は Levenshtein 符号を用いる場合のクロックモデルであり，提案する符号を用いる場合は，ビットアイランドとトレンチの幅が 1/2 になっているため，正規分布のパラメータの平均及び標準偏差をそれぞれ半分とする．また，τ^D 及び τ^b と τ^t を 2 及び 1 に正規化する．書き込まれたデータの各ビットは，確率 $P_{\rm bsc}$ で値が反転するとする．

4.2　シミュレーション結果

ランダムに生成した情報語を Levenshtein 符号及び提案した符号で符号化し，これを式 (2) 及び式 (3) に従って書き込んだ場合の復号誤り率を評価した．式 (2) における確率変数 ρ^D,ρ^b 及び ρ^t の標準偏差 σ^D,σ^b 及び σ^t が，$\sigma^D = 0, \sigma^b = \sigma^t$ で与えられるとした場合の復号誤り率を図 10 から図 12 に示す．ただし，図 10, 図 11 及び図 12 は，それぞれ反転誤り確率 $P_{\rm bsc}$ が $0, 10^{-5}$ 及び 10^{-4} の場合の復号誤り率を示す．Levenshtein 符号を

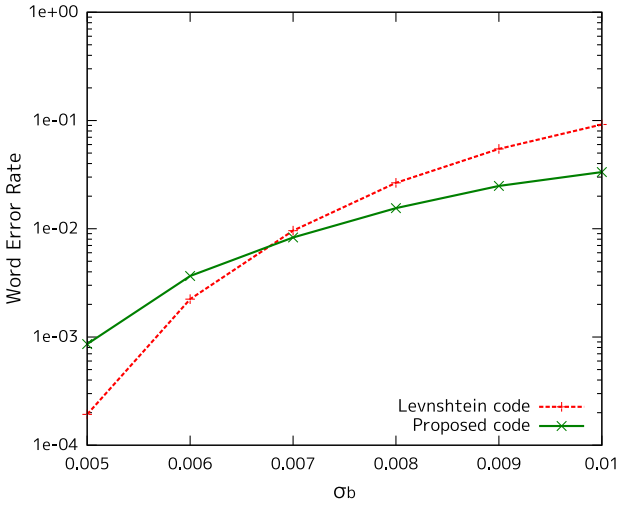

図 10　$P_{\rm bsc} = 0.0$ の場合の復号誤り率

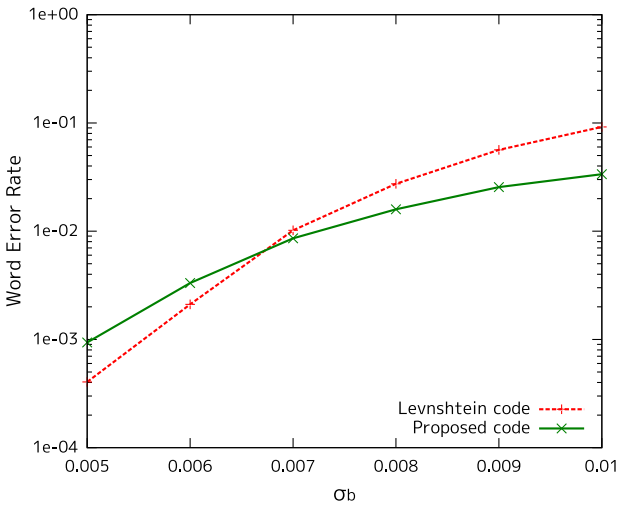

図 11　$P_{\rm bsc} = 10^{-5}$ の場合の復号誤り率

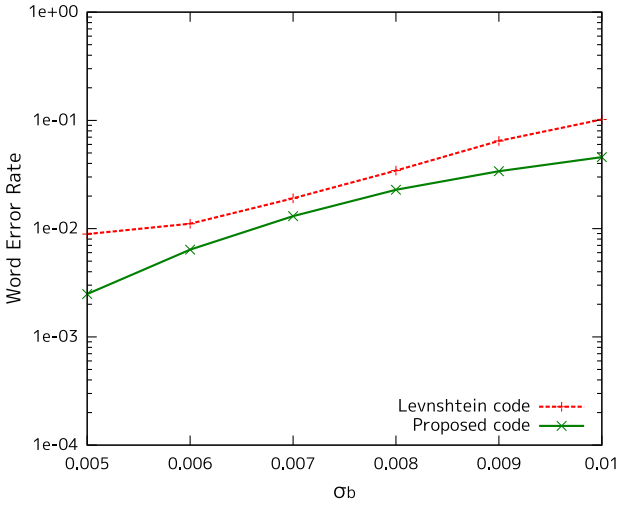

図 12　$P_{\rm bsc} = 10^{-4}$ の場合の復号誤り率

用いた場合と比較して，提案する符号を用いた場合は，$\sigma^b \geq 0.007$ の領域で復号誤り率は 1/2 程度まで低減する．

図 10 では，$\sigma^b = 0.005$ から 0.006 へ増加すると，Levenshtein 符号を用いた場合の復号誤り率が急激に増加する．Levenshtein 符号は単一削除／挿入／反転誤り訂正符号であるが，$\sigma^b = 0.006$ の付近で訂正能力を越える誤りが増加すると考えられる．図 12 に示すように，P_{bsc} が高く，反転誤りが同時に起きる場合は，提案する符号を用いると効果的である．

5 まとめ

本稿では，BPMR における同期ミスによる削除／挿入誤りを訂正するため，二重削除／挿入／反転誤り符号を提案した．シミュレーションによる評価の結果，既存の Levenshtein 符号を用いた場合と比較して，同期ミスと反転誤りが同時に起きる状況では，復号誤り率が 1/2 程度となる場合があることが明らかになった．

提案する符号により削除／挿入誤り確率が高い領域において，復号誤り率が Levenshtein 符号よりも低減したが，実用上は連接符号などにより復号誤り率をより低下させる必要がある．今後は，再生時の読み取りミスやビットアイランドのトラック間干渉などより現実的な条件での評価，Levenshtein 符号以外の削除／挿入誤り訂正符号，インターリーブを用いた場合との性能比較が必要である．また，本稿で提案した符号の符号化率は Levenshtein 符号と比較して 1/2 であるため，より良い符号化率の符号の考案が必要である．

参考文献

[1] K. Naito, H. Hieda, M. Sakurai, Y. Kamata, and K. Asakawa, “2.5-inch disk patterned media prepared by an artificially assisted self-assembling method,” *IEEE Trans. Magn.*, vol. 45, no. 2, pp. 822–827, 2009.

[2] Y. Tang, K. Moon, and H. J. Lee, “Write synchronization in bit-patterned media,” *IEEE Trans. Magn.*, vol. 47, no. 1, pp. 26–34, 2011.

[3] Y. Ng, B. V. K. V. Kumar, K. Cai, S. Nabavi, and T. C. Chong, “Picket-shift codes for bit-patterned media recording with insertion/deletion errors,” *IEEE Trans. Magn.*, vol. 46, no. 6, pp. 2268–2271, 2010.

[4] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” *Soviet Physics Doklady*, vol.10, no.8, pp.707–710, 1966.

[5] K. Saowapa, H. Kaneko, and E. Fujiwara, “Systematic Binary Deletion/Insertion Error Correcting Codes Capable of Correcting Random Bit Errors,” *IEICE Trans. Fundamentals*, vol. E83-A, no. 12, pp. 2699–2705, 2000.

[6] H. Muraoka and S. J. Greaves, “Statistical modeling of write error rates in bit patterned media for 10 Tb/in^2 recording,” *IEEE Trans. Magn.*, vol. 47, no. 1, pp. 26–34, 2011.