2005－IS－94（1）
2005／11／18

nビット黒化誤りに対するEANコードの信頼性

田口 敬教, 都倉 信樹
鳥取環境大学環境情報学部情報システム学科

概要 本稿では, 物品管理などに使われるEAN/UPC コードの信頼性についての検討結果を報告する. バーコードの黒化汚れによる誤読率をディジタル的に評価し，一ヶ所ｎビット黒化誤りに対しては, 誤読することは無いことが証明できた.

【キーワード】　EAN コード,n ビット黒化誤り, 信頼性

1　研究の目的・背景

EAN/UPC，以下簡単にEANと表す（日本ではこれをJANコードとも呼ぶ）は図1のように，黒と白の幅が4通りのバー (たての線）の組み合わせで13桁の数字列を表している．多くの商品に付けられているのでよく目にするものである．これは，国・メーカ・商品番号を表し, 最後の一桁はチェックディジットを表している．EAN コードは種々の情報システムで使われており，ＱＲコード，ＲＦＩＤ等と並び，自動認識技術の一つとして，個々の商品等の情報を迅速，確実に獲得する手段として今や必須のものとなっている．

図 1: EAN コード

実際にバーコードの付けられた商品を扱うコンビニ店の感覚では非常に読みとり精度がよい (誤読が少ない）というが，汚れやすい環境で使う場合バーコードが必ずしも精度よく読み取れないという声もある．

情報システムの設計者としては，入力の誤りの確率などを把握してシステム設計をするのは当然のことであると思われる．しかし，意外にもバーコードの読みとり精度や誤読率についてはほとんど情報がないという状況である．筆者の知る限り，唯一，文献 [4] に次の記述を見ただけである.「ＯＣＲ文字の誤読率はおよそ１万文字に１字で，バーコードは300万文字に1文字」(p.18)．EANのバーコードには，縦縞の黒白の模様のほかに，数字が書かれている．これはOCR（光学式文字読みとり装置）用の数字 (OCR-B フォント）を使っている．OCR 文字読みとりだと１万文字に１字程度誤読するが，バーコードでは，その300倍程度の好成績をあげられるというのである．

しかし，残念ながら，このデータがどのようにして得られたか，どういう使用状況でのデータなのかが不明である．これでは, 汚れの少ない環境と，比較的水物などを扱い汚れの付く可能性のある環境では，環境が違うのにこのデータをまるまる信用して使えないであろう．とくに，高信頼度が要求されるシステムでバーコードを使うべきか，まだ誤読率が高いRFID(ウォルマートの大規模フィールド実験では読みとり確率は90％程度でとても満足できないという報告もある) を選択すべきか，ここはシステム設計上大いに悩むところである．今後自動認識技術はバーコードしかないという状況から，いくつもの方法が選択できる時代に入り，どれを選択するのが合理的かの判断が常に求められるようになる．コストだけではだめで，本来の自動認識の目的である，人間に代わって機械的に読み取るというところでの性能の善し悪しがもっと議論されることになるであろう．バーコードは長い年月を経て実用化され，いま定着した技術であり，その誤読率などは十分データの蓄積があると期待していたが，公開されたものが乏しいことに驚いている．もし，情報をお持ちならご教示いただけると幸いである．

こういう状況で，ある適用環境でバーコードを使用できるかどうかを評価するために，誤読率を評価するにはどうすればよいかを検討し始めた．

ここで一つの思考実験を行う．実際に使う環境で，バーコードを実際の環境で使いどの程度誤読があるかを統計的に調べるのが当然もっとも確実な王道であろう．以下の説明は，この環境として，スーパーとして説明するが，別にそうでなくても本質は変わらない．

この実験システムを作るのが難しい．バーコードは使用する予定のバーコードリーダで読めばいいが，その読み取ったデータが正しいかどうかは，そのものの本当の番号を知らなければ照合できない．これを何によって与えるかが問題である．ＲＦＩＤとバーコードと二本立てでチェックするという案もある．ただし，ＲＦＩＤはコストが高いし，誤読率もかなり高いことを考えると，一体どちらをチェックしているのかということになりかねない．一つやや合理的かなと考える方法は，バーコードリーダで読み，オペレータがそのバーコードの下に記されているＯＣＲ文字を見る準備ができたときに，読み取った数字列を読み上げ装置で読んで，正しく読めているかをオペレータにチェッ

クしてもらうという方法である．実際，スーパーマーケットのオペレータは機械的に商品のバーコードを読ませているのではない．読み込んだ数字から知れる商品名や金額をつねにチェックしている．この人間のすばらしい能力を借りるという考え方である．しかし，オペレータの負担は少なくない．そして，1 アイテム当たりどのくらいの時間でこれが可能かは，商品の大きさや均一性などで影響されるが，仮に 20 秒と仮定してみる．精算や包装等別の作業や休憩，客の到着も粗密があり，営業時間中 20 秒／アイテムというのは過小評価かも知れない．それでも，一日 12 時間営業の店舗で，一つの実験レジで，$3 \times 60 \times 12 = 2160$ アイテムのデータがとれる．1 年無休で，788,400 アイテムのデータになる．一つのバーコードは 13 文字なので，これで 10, 249,200 文字を読んだことになる．これで 3 文字誤読があったとわかったとき，先のデータ約 300 万字に 1 回の誤読と主張できるだろうか．おそらくデータが十分でないと批判されるのではないだろうか．

また，この実験のコストはどう見積もればよいであろうか．通常作業の中でできるからコスト zero と言っていいのだろうか．そうではない．読み上げシステムの追加や，オペレータの訓練や，どうしても OCR 文字との照合でレジはスムーズに流れなくなり，レジの生産性が落ちることなどを考慮しなければならない．もちろん，複数のレジを使って同時にやれば，短期間でデータは集められるが，総コストはあまり変わらない．システム提案前にこのような実験をするというふうに踏み切れるだろうか．

実際の状況でデータを取ることは，この例に限らないが一般になかなか難しいのは事実である．そして，少なからずコストを投じて誤読率のデータを得ても，別の環境では，誤りのパターンが違えば，そのまま適用することはできないであろう．

そこで，近似的な方法というべきかも知れないが，第 2 の方法を考えた．本論文はその新しいアプローチを紹介し，第一段階の結論を報告するものである．

本論に入る前にいくつか用語を定義する．

2　用語説明

2.1　モジュール

EAN コードでは，1 モジュールは EAN コードの中の一番細い黒または白の線の幅である.EAN コードでは全部で 95 モジュールで表されている.(図 2)

2.2　ビット

1 モジュールは黒と白で表されるのに対し,1 つの黒モジュールを 1，白モジュールを 0 と表した場合に 1 ビットと読み替える．(図 2)

図 2: EAN のモジュール

2.3　バー

EAN コードの連続した黒い部分を黒バー，連続した白い部分を白バーとよぶ. 黒バー・白バーはそれぞれ 1〜4 モジュールの幅が許されている．(図 2)

2.4　黒化

黒化とは, バーコードに黒い汚れが付くこと，およびその結果黒く見えることをいう.(図 3) なお，反対に白化も考えられるであろうが，本文は黒化のみを扱う.

2.5　黒化誤り

黒化誤りとは, 黒化した部分を黒バーと認識する場合に黒化誤りとよぶ. (図 3)

2.6　失読

バーコードリーダは 1 秒間に数百回という多方向に高速スキャンをしており，うまく読めたときにピッという音で読めたことをオペレータに通知する．正しく読めるまで，スキャンの方向や傾き，すべての線を横切っていない等，種々の状況で，正しくバーコードが読めないとき，その試行は失読したといい，再スキャンを行う.

2.7　誤読

なんらかの汚れや屈折異常などで，本来の数字と違う数字に誤ってよみ，正常に認識したと扱う (リーダーはピッと通知する）場合を誤読とよぶ. 問題は失読でなく，誤読である．これは取引情報に誤りを持ち込むことになり，あってはならないことである．この誤読の確率を評価することが本報告の主題である.

2.8　n ビット黒化誤り

1 モジュールが黒化する事を 1 ビット黒化誤りといい,n モジュールが黒化する事を n ビット黒化誤りという

3　誤読率の第 2 の評価法

このアプローチは後から考えると，符号理論のアプローチに似ている．符号理論では，実際の通信路での変復調や通信路での誤りを直接扱うことをさけ，0 と 1 の 2 進の世界に変換して理論化している．通信路は

図 3: 黒化

２元対称通信路や，非対称通信路としてモデル化されたり，バースト誤りを想定してそれに耐える符号を設計するというアプローチを行って，極めて豊かな理論的成果を上げ，今日それが QR コードや，DVD などにも利用されるようになっている．ややそれに似て，実際はバーコードをレーザ光でスキャンしたとき得られる波形は当然アナログであるが，これを適当に量子化してバーコードの規約に照らして復号しているのである．本報告のアプローチは，黒い汚れがあったとして，それは白い０ビットが黒を表すビット１に変わるというモデル化を行う．そして，そういう誤りがあったとき，失読になって再スキャンすることになるのか，そのまま間違って読み込んで誤読になるのかを検討するという方針である．

今回の報告では，想定した誤りは，１ヶ所に１ビットの黒化誤りが起こる場合，１ヶ所であるが，２ビット，３ビット，４ビット,... の長さに誤る場合を想定している．それはバーコードのサイズは緩く規定されており，モジュールが小さいバーコードもあれば，かなり大きいバーコードもある．同じ１滴の水による汚れでも，小さいバーコードであれば，それは数ビットの誤りになるかもしれないが，大きなバーコードだと１ビットの誤りになる．したがって，通信路の場合と違った誤りの解釈が求められる．今後，実際にどういう誤りが，０，１の世界でのどの誤りパターンに結びつくかの調査をしなければならない．誤りのパターンを分類し，それを E i(i=1,...,m) とする．現場のバーコードの汚れが，E i に分類されるものとなる確率が p i と求められ，０，１の世界の解析で誤りパターン E i の場合の誤読率 q i が求めておけば，

$$\sum_{i=1}^{m} piqi$$

で，誤読率をもとめることができる．q i は今回の報告に一部の誤りパターンについて求めているが，残りのパターンについてその値を計算することはできる．これに対し，p i は実際に使いたい環境で，バーコードの種々のサンプルを収集し，それがどの誤りパターンにクラスに対応するかを統計的に求めればよい．このコストは最初のアプローチに比較し，かなり低いと想定している．少なくとも，実際に使用する状況でデータを取らなくても，その現場におかれたバーコードを多数収集し，一定の方法で操作してどの誤りパターンになるかを調べるシステムを作ればよい．この手法については，今後の検討課題である．まずは，q の方のデータを理論的に求めておくことが第一ステップであると考えている．

次に，その具体的方法に話を進める．

	左側データキャラクタ		右側データキャラクタ
10進数字			チェックディジット
	奇数パリテ	偶数パリテ	偶数パリティ
0	0001101	0100111	1110010
1	0011001	0110011	1100110
2	0010011	0011011	1101100
3	0111101	0100001	1000010
4	0100011	0011101	1011100
5	0110001	0111001	1001110
6	0101111	0000101	1010000
7	0111011	0010001	1000100
8	0110111	0001001	1001000
9	0001011	0010111	1110100

図 4: EAN のコード

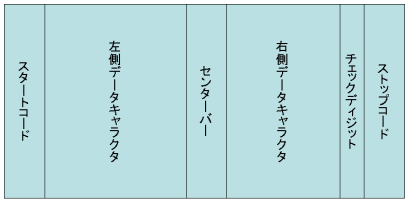

図 5: EAN の構造

4　EAN コードとは

4.1　モジュールの数

図 5 のように EAN コードでは, スタートコード，センタバー，ストップコードがあり, それぞれのコードの間に, 左側データキャラクタ，右側データキャラクタがある. スタートコードとストップコードは黒，白，黒の 3 モジュールで表される. センターバーは白，黒，白，黒，白の 5 モジュールで表される. データキャラクタとは, バーコードの下についている数字のことである．一文字のデータキャラクタはそれぞれ 7 モジュールで表現される.(図 2)

左側データキャラクタは 6 文字の数字を表し, 全 95 モジュールで表され, このモジュール数はどの EAN コードも同一である.

4.2　偶奇パリティ・プリフィックスキャラクタ

左側データキャラクタは奇数パリティのものと偶数パリティのものからなる.(図 4) 奇数パリティの キャラクタとは黒バーのモジュール数が奇数個（3 か 5）のものである．偶数パリティのキャラクタとは黒バーのモジュール数が偶数個（2 か 4）のものである．EAN コードでは 13 桁の数字の列が表現されているが，デー

プリフィックス文字	O（奇数）とE（偶数）の順	プリフィックス文字	OとEの順
0	OOOOOO	5	OEEOOE
1	OOEOEE	6	OEEEOO
2	OOEEOE	7	OEOEOE
3	OOEEEO	8	OEOEEO
4	OEOOEE	9	OEEOEO

図 6: EAN のプリフィックスキャラクタの算出方法

	0							結果
0001101	0	0	0	1	1	0	1	
1001101	1	0	0	1	1	0	1	変化あり
0101101	0	1	0	1	1	0	1	変化あり
0011101	0	0	1	1	1	0	1	変化あり
0001101	0	0	0	1	1	0	1	変化なし
0001101	0	0	0	1	1	0	1	変化なし
0001111	0	0	0	1	1	1	1	変化あり
0001101	0	0	0	1	1	0	1	変化なし

図 7: 検証

タキャラクタで表現される文字数は 6+5+1 の 12 文字しかない. その秘密は，図 6 のような偶奇パリティの並び方で左端の数字を表すことにある．３．の左データキャラクタは奇数パリティのキャラクタの総数が 3 つか 6 つということは図 6 の表よりわかる. バーコードを左から読んでも, 右から読んでも正しく認識できるのは, 右側データキャラクタおよびチェックディジットは偶数パリティのものしか存在しないため, どちらが左側データキャラクタで右側データキャラクタであるかを区別できるからである.

4.3 データキャラクタとは

図 4 のように定義されている. 左側データキャラクタは 1 ビット目が 1 で 7 ビット目が 0 で, 右側データキャラクタは 1 ビット目が 0 で 7 ビット目が 1 である. 偶数パリティキャラクタの場合, 一文字に付き 1 が 2 つか 4 つある. 対して奇数パリティキャラクタの場合は一文字に付き 1 が 3 つか 6 つある. そして, 一つのデータキャラクタには黒バーと白バーの数が二つずつある. それぞれ左側データキャラクタは白，黒，白，黒という並びで, 右側データキャラクタは黒，白，黒，白である.EAN コードでは黒バーの数は 6 × 2 × 2 + 2 + 2 + 2 で 30 カ所, 白バーの数は 6 × 2 × 2 + 2 + 2 + 1 の 29 カ所で定義されている.

4.4 チェックディジット

EAN コードにはチェックディジットが含まれている.EAN でのチェックディジットは 12 桁のデータに対し, 左から a, b, c, d, e, f, g, h, j, k, m, n, p とすると, 正しいバーコードでは

$a+c+e+g+j+m+p+3(b+d+f+h+k+n) = 0(mod10)$

となっていなければならない」と, 定められている.

5 EAN コードの評価

5.1 1 ビット黒化

図 7 と図 8 の場合を説明する. ○ 1 ビット目

左側データキャラクタの 1 ビット目は白という定義がある. そのため,1 ビット目が黒バーになる場合,EAN コードの定義からはずれ失読になる.

○ 2 ビット目

黒バーの数が 3 カ所になるため,EAN コードの定義からはずれ失読. 黒バーの数が 3 カ所になるのは黒化誤りを発生する場所が 0$\underline{0}$0 のように 0 に囲まれている場合である.

○ 3 ビット目

黒バー，白バーの数は共に 2 カ所のため, バーの数では検出できない. 0 が 1 に変化すると, パリティが変化する. 元々奇数パリティの 0 のパリティが変化することによって, 偶数パリティキャラクタになる. EAN コードは奇数パリティキャラクタが 3 つか 6 つという定義からはずれるため, 黒化誤りを検出できる

○ 4,5,7 ビット目

元々 1 のため, 黒化誤りは発生しない

○ 6 ビット目

101 の部分が変化するため黒バー，白バーの数ともに減り,1 となる.EAN コードの定義からはずれるため, 黒化誤りを検出できる.

一般の場合

○左側データキャラクタ

1 ビット目に黒化が発生した場合, 必ず黒化誤りを検出できる．7 ビット目に黒化が発生した場合, 黒化誤りはあり得ない．２～６ビット目のいずれかに黒化が発生した場合, 0$\underline{0}$0 が 0$\underline{1}$0 と変化した場合, 黒バーの数が増えるので黒化誤りを検出できる．1$\underline{0}$0 と 0$\underline{0}$1 が 1$\underline{1}$0 と 0$\underline{1}$1 に変化した場合, パリティが変化するため, 黒化誤りを検出できる．1$\underline{0}$1 が 1$\underline{1}$1 に変化した場合, 黒バー・白バーの数共に減り, 黒化誤りを検出できる．黒化が発生した場所が 1 の場合, 元々1 のため黒化誤りとならない.

○右側データキャラクタ

1 ビット目もともと１）に黒化が発生した場合, 黒化誤りはあり得ない. 7 ビット目に黒化が発生した場合, 必ず黒化誤りを検出できる．２～６ビット目のいずれかに黒化が発生した場合, 左側データキャラクタと同様の結果になる.

5.2 2bit 黒化

2 ビット黒化の場合,2 ビットの連続したバーが黒くなる. つまり,1-2 ビット目,2-3 ビット目,3-4 ビット目,4-5 ビット目,5-6 ビット目,6-7 ビット目の 6 パターンを考えればいい.

- 1-2 ビット目の黒化

  左側データキャラクタの黒化に対しては, 1 ビットの時と同様必ず先頭のビットは 0 でなければな

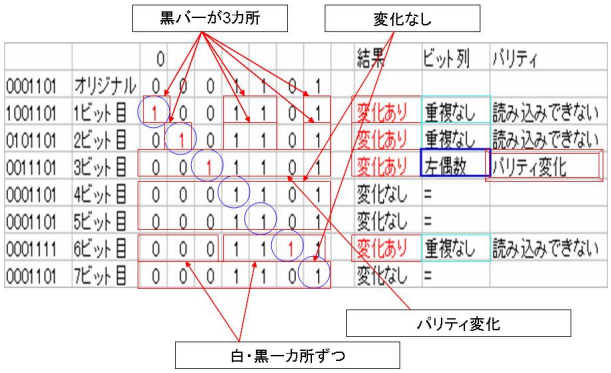

図 8: 検証後

らないため, 黒化誤りを判定できる. 右側データキャラクタに対しては, 1 ビット目が必ず 1 のため, 黒化が起きても, 変化するのは 2 ビット目だけである. つまり,2 ビット連続黒化が起きても, 実質 1 ビット黒化誤りと同じとみなされるので,1 ビット黒化誤りの時と同様, 判定できる.

- 6-7 ビット目の黒化

  左側データキャラクタの黒化に対しては, 1 ビットの時と同様 7 ビット目は必ず 1 でなければならないため, 2 ビット連続黒化が起きても, 実質 1 ビット黒化誤りと同様である. そのため,1 ビット黒化誤りの時と同様, 判定できる. 右側データキャラクタに対しては,7 ビット目が必ず 0 でなければならないため, 判定できる.

- 1 が連続した部分の黒化

  黒化が発生した場合, 黒化誤り自体は判定できないが, 元々1 なので, 見かけのパターンの差は発生せず, 問題ない.

  上記より,1-2 ビット目,6-7 ビット目,1 の連続したビットを除く, 00,01,10 の部分だけに起きる黒化だけを考えればよい.

- 01 の部分に黒化が起きた場合

  この場合も,2 ビット連続黒化が起きても, 実質 1 ビット黒化誤りと同様である.

- 10 の部分に黒化が起きた場合

  この場合も,2 ビット連続黒化が起きても, 実質 1 ビット黒化誤りと同様である.

- 1001,1001 の部分に黒化が起きた場合

  白バーと黒バーの数が減るので, 全体の白バー, 黒バーの数が 1 ずつ減り判定できる.

- 変化する 0 のとなりが 0 の場合（10001,10001 など）の 0 に黒化が発生すると黒バー, 白バーの数ではチェックできず, パリティでもエラーを検出できない.

  この場合, 元々のデータキャラクタを x, 黒化によって発生した黒化誤りで, 変化した数字を y,x と y の差を z とすると,

  $x \neq y$, $z \neq z$ となる.（$x = y$ の場合, 黒化による影響は無いとされるため）

  4.4 より, $mod10$ の計算結果が $|z|$ もしくは $|3z|$ となるため, データキャラクタが単一カ所変化することによって起きる誤りは検出できる.

上記より,2 ビット黒化においても黒化誤りの発生を判定でき, 誤読することはない.

5.3 3 ビット黒化の結果

3 ビット黒化の場合, 黒化が起こることにより,3 ビットの連続したバーが黒くなる. つまり,1-2-3 ビット目,2-3-4 ビット目,3-4-5 ビット目,4-5-6 ビット目,5-6-7 ビット目の 5 パターンを考えればいい.

- 1-2-3 ビット目の黒化

  左側データキャラクタの黒化に対しては, 必ず先頭のビットは 0 でなければならないため, 黒化誤りを判定できる. 右側データキャラクタに対しては, 1 ビット目が必ず 1 のため, 黒化が起きても, 変化するのは 2-3 ビット目だけである. つまり,3 ビット連続黒化が起きても, 実質 2 ビット黒化誤りと同じとみなされるので,2 ビット黒化誤りの時と同様, 判定できる.

- 5-6-7 ビット目の黒化

  左側データキャラクタの黒化に対しては, 7 ビット目は必ず 1 でなければならないため, 3 ビット連続黒化が起きても, 実質 2 ビット黒化誤りと同様である. そのため,2 ビット黒化誤りの時と同様, 判定できる. 右側データキャラクタに対しては,7 ビット目が必ず 0 でなければならないため, 判定できる.

- 1 が連続した部分の黒化

  黒化が発生した場合, 黒化誤り自体は判定できないが, 元々1 なので, 見かけのパターンの差は発生せず, 問題ない.

  上記より,1-2-3 ビット目,5-6-7 ビット目,1 の連続したビットを除く, 110,011,000 の部分だけに起きる黒化だけを考えればよい.

- 110,011 の部分に黒化が起きた場合

  この場合, 事実上, 元と違うのは 0 の部分だけとなり, 1 ビット黒化誤りの場合と同様になる.

- 100,001 の部分に黒化が起きた場合

  この場合も, 事実上, 元と違うのは 0 の部分だけなので 2bit 黒化誤りの場合と同様になる.

- 000 の部分に黒化が起きた場合

  3bit 変化することによって, いずれのキャラクタもパリティが変化するため, プリフィックスキャラクタが変わるので誤読することはない.

上記より,3 ビット黒化においても黒化誤りの発生を判定でき, 誤読することはない

5.4 4 ビット黒化の結果

4 ビット黒化の場合, 黒化が起こることにより,4 ビットの連続したバーが黒くなる. つまり,1-2-3-4 ビット目,2-3-4-5 ビット目,3-4-5-6 ビット目,4-5-6-7 ビット目の 4 パターンを考えればいい.

- 1-2-3-4 ビット目の黒化

  1,2,3 ビット黒化と同様, 左側データキャラクタの黒化に対しては, 必ず先頭のビットは 0 でなければならないため, 黒化を判定できる. 右側データキャラクタに対しては, 1 ビット目が必ず 1 のため, 黒化が起きても, 変化するのは 2-3-4 ビット目だけである. つまり,4 ビット連続黒化が起きても, 実質 3 ビット黒化誤りと同じとみなされるので,3 ビット黒化誤りの時と同様, 判定できる.

- 4-5-6-7 ビット目の黒化

  左側データキャラクタの黒化に対しては, 7 ビット目は必ず 1 でなければならないため, 4 ビット連続黒化が起きても, 実質 3 ビット黒化誤りと同様である. そのため,3 ビット黒化誤りの時と同様, 判定できる. 右側データキャラクタに対しては,7 ビット目が必ず 0 でなければならないため, 判定できる.

- 1 が連続した部分の黒化

  黒化が発生した場合, 黒化誤り自体は判定できないが, 元々1 なので, 見かけのパターンの差は発生せず, 問題ない.

  上記より,1-2-3-4 ビット目,4-5-6-7 ビット目,1 の連続したビットを除く, 0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001, 1010, 1011, 1100, 1101, 1110 の部分だけに起きる黒化だけを考えればよい.

- 0111,1011,1101,1110 の部分に黒化が起きた場合

  この場合, 事実上, 元と違うのは 0 の部分だけとなり, 1bit 黒化誤りの場合と同様になる.

- 0011,0101,0110,1001,1010,1100 の部分に黒化が起きた場合

  この場合も, 事実上, 元と違うのは 0 の部分だけなので 2bit 黒化誤りの場合と同様になる.

- 0001,0010,0100,1000 の部分に黒化が起きた場合

  この場合も, 事実上, 元と違うのは 0 の部分だけなので 3bit 黒化誤りの場合と同様になる.

- 0000 の部分に黒化誤りが起きた場合

  4bit 変化することによって, いずれのキャラクタも白バーの数が減るため, バーコードとして認識しなくなる.

上記より,4 ビット黒化においても黒化誤りの発生を判定でき, 誤読することはない.

5.5 5bit 以上連続の黒化

データキャラクタでは, 最高 4bit しか 0 が連続していないため, 5bit 以上の黒化が起きることによって, 白バーの数が減り, バーコードとして認識しなくなる. よって,5bit 以上連続の黒化で誤読することはない.

5.6 nbit 黒化の結果

上記より,1 ビット,2 ビット,3 ビット,4 ビット,5 ビット,6 ビット,7 ビットで黒化が発生しないため, 黒化が発生しても, 誤読することはない.

5.7 複数データキャラクタにまたがる黒化

隣り合う二つのデータキャラクタにまたがる黒化の場合, 黒化が起こったデータキャラクタの左側は 7 ビット目, 右側は 1 ビット目が黒化するため, 黒化が起きても黒化誤りを判定でき, 誤読することはない

6 おわりに

本稿で,n ビット黒化誤りにおける誤読は無いことを証明した. 黒化誤りがモジュールが黒バーに変化するのに対し, 白バーに変化する白化誤りを検証し, 別のバーコードも検証して行きたいと思う.

参考文献

[1] 都倉信樹,”情報論 05”, 鳥取環境大学 2005 年度用テキスト,2005 年

[2] 小塚洋司,”バーコードの秘密”, 裳華房,1996 年

[3] 財団法人 流通システムセンター,”JAN シンボルによるソースマーキングマニュアル”,1992 年

[4] 平本純也, 知っておきたいバーコード,2 次元コードの知識, 日本工業出版,1991 年