目 次

第1章　序論

QR コード [1] は、1994 年に株式会社デンソーウェーブが開発した二次元バーコード であり、食品や製造業の在庫管理など多方面の分野で利用されている．一般的な QR コードは、白と黒の正方形のモジュールで構成されておりデザイン性を考慮していない. 一方で広告、サービス業界ではデザイン性を考慮した QR コードが求められている．デザイン性を考慮した QR コードでは、一定のルールから QR コードを変更することによって、QR コード上にロゴ画像（以後、目的画像と述べる）を埋め込んだものがある．このような QR コードを Aesthetic QR コード [3] という．

Aesthetic QR コードの研究は大きく三種類に分けることができる:

1. QR コードの一部に目的画像を埋め込む方法,
2. 画像のヒストグラムを考慮して目的画像を埋め込む方法 [2],
3. QR コードに利用されている RS 符号中の padding codewords と呼ばれる領域 (以下、埋め草コード語) を考慮して目的画像を埋め込む方法 [3].

本研究は、上に述べた三つ目に分類される埋め草コード語を考慮して目的画像を埋め込む方法を考察し、目的画像に近い Aesthetic QR コードを自動生成する方法について検討する．Aesthetic QR コードを自動生成する手法として、本研究では、Kuribayashi らの論文 [3] で提案されている Ramdom Method のアルゴリズムを用いる.QR コードで用いられる RS 符号の計算は代数拡大体上で計算されるため、本研究では数式処理システム Maple を用いて Aesthetic QR コードを生成するソフトウェアを開発した．

以下、第 2 章では QR コードを構成する Reed-Solomon 符号と QR コードの概要について述べ、第 3 章では本研究で用いる Kuribayashi らの論文 [3] の Ramdom Method について述べる. 第 4 章では数式処理 Maple による AesthiticQR コードの実装とその結果について述べる. 第 5 章では結論と今後の課題について述べる.

第2章　QRコード

この章では AestheticQRcode をの構成要素である Reed-Solomon 符号（以下、RS 符号）と QR コードについて述べる.

2.1　RS 符号

RS 符号とは符号理論における誤り訂正符号の一つである. その高い誤り訂正能力から、QR コードなどに応用されている.

以下に、RS 符号の各用語について説明する.

1. 符号多項式

   符号長 n の線形符号 C の任意の符号語をベクトル表現

   $$v = (v_0, v_1, v_2, \cdots, v_{n-1}) \tag{2.1}$$

   としたとき式 2.1 の多項式表現は

   $$v(x) = v_0 + v_1 x + v_2 x^2 + \cdots + v_{n-1} x^{n-1} \tag{2.2}$$

   である. ここで変数 x^i は単に記号 v_i の位置を示すだけである. 式 2.2 のようにある符号語に対応する多項式を特に符号多項式と呼ぶ.

2. 生成多項式

   ある情報記号と対応する多項式 (以下、情報多項式) $q(x)$ からこれの誤り訂正を行う符号語 $u(x)$ を生成することを考えた際

   $$v(x) = q(x)g(x) \tag{2.3}$$

   と表される $g(x)$ を生成多項式と呼ぶ.

3. 体

体とは代数学においてある性質を満たした集合である. 体の性質の中でも最も特徴的な点は、元 (体の要素をこのように呼ぶ) の四則演算は結果も元になる (体の中で閉じている) という点である. 例えば実数は体であり、実数を用いた四則演算は計算結果が全て実数になる. しかし自然数は体ではなく、例えば 1 – 2 の演算結果は負の値となりこれは自然数ではないのでこれは体とは言えない.

4. ガロア体 (有限体)

体の中でも元が有限なものをガロア体 (Galois field) と呼び、有限体とも呼ばれる. 元の数が q のガロア体を $GF(q)$ で表し、元の数は素数、あるいは素数のべき乗である必要がある. 例えば $GF(2)$ の元は一般的に 0 と 1 である. 計算例として以下に $GF(2)$ 上の加算減算の計算結果を示す.

表 2.1　GF(2) 上の加算結果

入力 1	入力 2	出力
0	0	0
0	1	1
1	0	1
1	1	0

表 2.2　GF(2) 上の減算結果

入力 1	入力 2	出力
0	0	0
0	1	1
1	0	1
1	1	0

5. 拡大体, 原始元, 原始多項式

ガロア体 $GF(p)$ 上の既約多項式 (これ以上因数分解できない多項式) $g(x)$ を選び、その根 ($g(x) = 0$ となるような x の値) を α とする. この α を $GF(p)$ の元に追加することで新たな体が生成でき、そうしてできた新たな体を拡大体と呼ぶ. この時の α を原始元といい、この既約多項式は原始多項式という. 拡大体の例として複素数が挙げられる. 複素数は実数の拡大体であり実数上の既約多項式 $x^2 + 1$ の根を i として実数の元に追加したものである.

拡大体 $GF(2^m)$ 上の RS 符号について考える. 拡大体 $GF(2^m)$ の原始元を α とするとき、$\alpha, \alpha^2, \alpha^3, \cdots, \alpha^{2t}$ を根として持つ $GF(2^m)$ 上の生成多項式 $g(x)$ により生成される符号を t 重誤り訂正 RS 符号と呼ぶ.

例えば RS(26,16) 符号について考える. 入力情報を

$$\alpha^6, \alpha^{110}, \alpha^{48}, \alpha^{239}, \alpha^{99}, \alpha^{129}, \alpha^{15}, \alpha^4, \alpha^{125}, \alpha^{68}, \alpha^{57}, \alpha^{78}, \alpha^{49}, \alpha^{127}, \alpha^{127}, \alpha^{127} \tag{2.4}$$

とし、これを多項式表現したものと生成多項式

$$x^{10} + \alpha^{251}x^9 + \alpha^{67}x^8 + \alpha^{46}x^7 + \alpha^{61}x^6 + \alpha^{118}x^5 + \alpha^{70}x^4 + \alpha^{64}x^3 + \alpha^{94}x^2 + \alpha^{32}x + \alpha^{45} \quad (2.5)$$

より生成された生成行列 G をかけ合わせることで、符号多項式

$$x^{10} + \alpha^{251}x^9 + \alpha^{67}x^8 + \alpha^{46}x^7 + \alpha^{61}x^6 + \alpha^{118}x^5 + \alpha^{70}x^4 + \alpha^{64}x^3 + \alpha^{94}x^2 + \alpha^{32}x + \alpha^{45} \quad (2.6)$$

が生成される.

2.2　QR コードの概要

QR コードの構成要素の最小単位は白と黒で表されるモジュールであり、白のモジュールは 0 のビット値を表し、黒のモジュールは 1 のビット値を表す. QR コードのサイズはバージョンによって決定され、そのバージョン (v) は 1 ~ 40 である. 1 型は、21 × 21 モジュール、2 型は、25 × 25 モジュール、というように、型番が一つ上がるごとに一辺につき 4 モジュールずつ増加し、40 型は、177 × 177 モジュールとなる. したがって、バージョン v は $(17 + 4v) \times (17 + 4v)$ モジュールである.

図 2.1 に QR コードの構成要素を表す.

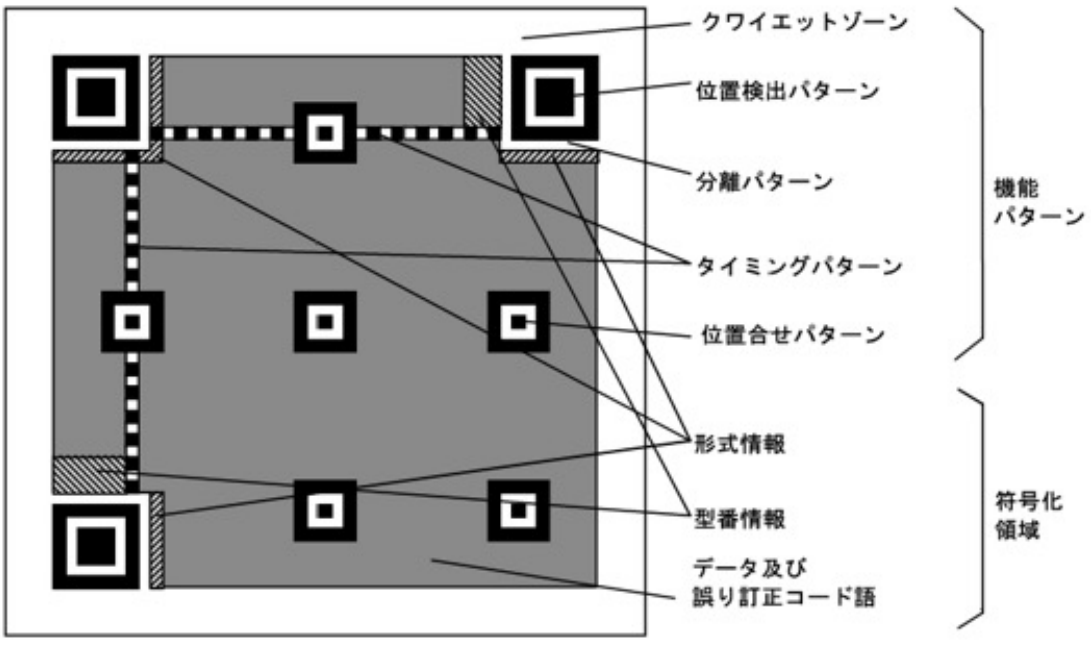

図 2.1　QR コードシンボルの構造 [7]

QR コードは、QR コード上にある符号化されたデータを正確に認識するために機能パターンを持っている. 機能パターンは主に 3 つの構成要素から成り立っており、それぞれ位置検出パターン、位置合わせパターン、タイミングパターンと呼ばれる.

位置検出パターンは QR コードの左上、左下、右上の角にある 3 つの正方形のブロックである. それらの境目を明白にするために、形式情報との間に白いモジュールを置く. これを分離パターンと呼ぶ.

位置合わせパターンは小さな正方形のブロックで、位置検出パターンの垂直・平行座標に関係する位置に置く. バージョンによっては付加しない場合もあり、バージョン 1 には存在しない.

タイミングパターンは左上の位置検出パターンから右上の位置検出パターンへと、左上の位置検出パターンから左下の位置検出パターンへの白黒が交互に並ぶ 2 つのラインのことである.

QR コードのデータビットは、QR コードの右下から始まり、2 モジュール幅の列上に配置する．列が最上部に達すると、次の 2 モジュール列は右端から始まり、下方向へ続く．現在の列が端に達すると、次の 2 モジュールの列に移動して方向を変更する．データビットは機能パターン（位置検出パターン、タイミングパターン、位置合わせパターン）の位置では、次のモジュールへ配置される.

上方向のデータビットの配置は図 2.2 に、下方向のデータビットの配置は図 2.3 に示す.

0	1
2	3
4	5
6	7

図 2.2　上方向のビット配列

6	7
4	5
2	3
0	1

図 2.3　下方向のビット配列

またデータの格納方法にも種類があり、英数字モードや 8 ビットバイトモードなどがある.

QR コードは、誤り訂正符号として RS 符号を使用し、その能力は L、M、Q、H の 4 つ

のレベルに昇順で分類される. 各誤り訂正レベルは QR コード内の全シンボルの約 7%、約 15%、約 25%、約 30% までのシンボルを訂正することができる. それぞれを表 2.3 に示す.

表 2.3　誤り訂正レベル

レベル	L	M	Q	H
誤り訂正能力	約 7%	約 15%	約 25%	約 30%

2.2.1　QR コード上の RS 符号

以下に QR コード上での RS 符号について示す。

RS 符号の各シンボルは拡大体 $GF(2^8)$ を使用し、原始多項式は $x^8 + x^4 + x^3 + x^2 + 1$ を使用する. この原始多項式の原始元 α を用いて計算を行う.

RS 符号の情報長は固定されているが、QR コードに入れる情報が少ない場合、埋め草コード語が追加される. 例えばバージョン 1 の QR コードを構成する RS 符号の長さは 26 シンボルとなっている. そのうちの情報記号の長さは 16 シンボルである.QR コードに入れる文字列が 16 シンボル未満で表現される場合、情報記号のシンボル数を 16 シンボルに合わせるため、埋め草コード語という情報を持たないシンボルを付加する必要がある．QR コードに書き込む文字列を RS 符号化した際のシンボル (以下、データと述べる) の個数を $\hat{k}$ と表すとき、データを表す RS 符号のシンボルは

$$\alpha_1, \cdots, \alpha_{\hat{k}} \tag{2.7}$$

である．$\hat{k} < k$ の時、RS 符号のデータを表すシンボルを k 個に合わせるため、埋め草コード語を付加する．これにより、RS 符号の情報記号のシンボルは

$$\alpha_1, \cdots, \alpha_{\hat{k}}, \alpha_{\hat{k}+1}, \cdots, \alpha_k$$

と表す．なお、埋め草コード語は

$$\alpha_{\hat{k}+1}, \cdots, \alpha_k \tag{2.8}$$

と表す.

QR コード上の RS 符号の全体図を図 2.4 に示す.

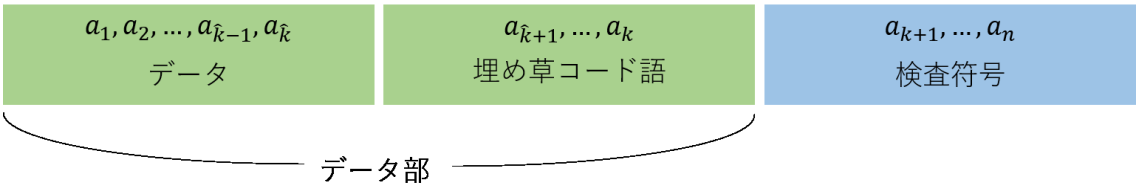

図 2.4　QR コード上の RS 符号の全体図

QR コード上の生成多項式 [5] [6] は $n-k$ 次多項式であり、これを $g(x)$ とする.

$$g(x) = (x-1)(x-\alpha)\cdots(x-\alpha^{n-k-1}) = \prod_{i=0}^{n-k-1}(x-\alpha^i) \tag{2.9}$$

式 2.9 の $g(x)$ を展開した多項式を

$$g(x) = g_1 x^{n-k} + g_2 x^{n-k-1} + \cdots + g_{n-k+1},\ g_1 = 1 \tag{2.10}$$

とする. その係数列 $g_1 = 1, g_2, \cdots, g_{n-k1}$ から定まる $k \times n$ 行列

$$G_0 = \begin{bmatrix} 1 & g_2 & \cdots & g_{n-k-1} & 0 & \cdots & \cdots & 0 \\ 0 & 1 & g_2 & \cdots & g_{n-k-1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & g_2 & \cdots & g_{n-k-1} & 0 \\ 0 & \cdots & \cdots & 0 & 1 & g_2 & \cdots & g_{n-k-1} \end{bmatrix}$$

を生成行列と呼ぶ.

一般的に QR コードでは情報記号と検査記号を分けた組織符号が用いられる. 行列 G_0 を使って組織符号を計算するためには、行列 G_0 を”掃き出し法”によって $G = [I_k, P]$ (I_k は $k \times k$ 単位行列) という標準形に変形する.

データ u に対する符号語 v は標準形の生成行列 $G = [I_k, P]$ により、

$$v = uG \tag{2.11}$$

として表現することができ、この v が QR コード上の RS 符号である。

第3章　Aesthetic QRコード

本章では、Kuribayashi らの論文 [3] の Random-Method を述べる.Random-Method は QR コード上に配置された二値配列が埋め込む目的画像のモジュールパターンと類似した Aesthetic QR コードを生成する. 本研究では QR コードに埋め込む画像を目的画像と呼ぶ．Aesthetic QR コードの例を図 3.1 に示す.

図 3.1　Asethetic QR コード例

3.1　色変換手法

目的画像はカラー画像でも二値画像でもよい. カラー画像や二値画像を用いる際に、目的画像と QR コードとの距離を測るために、目的画像を二値化する必要がある. ここでの距離にはハミング距離を用い、最もハミング距離の小さい QR コードを用いて Aesthetic QR コードを作る. 目的画像を二値化画像にする際に閾値を指定するが、その閾値を Algorithm 1 に示す色変換手法で決定する.

閾値を用いて目的画像から目的画像の二値行列を作成するアルゴリズムを Algorithm 2 に示す.

Algorithm 1 論文 [3] の色変換手法

入力: サイズ $L \times L$ の目的画像

出力: 閾値 $\overline{Y}$

方法:
1. 入力画像の大きさは、QR コードのバージョン v と同じサイズに予め変更しておく．RGB 色成分は YUV 色成分に変換され、輝度 (Y) 成分 $Y_{i,j}$,$(1 \leq i, j \leq L)$ が得られる．
2. その中心の正方形 (元の画像サイズの $\frac{1}{4}$) の値の平均 $\overline{Y}$ が計算する．

$$\overline{Y} = \frac{4}{L^2} \sum_{i=\frac{L}{4}}^{\frac{3L}{4}-1} \sum_{j=\frac{L}{4}}^{\frac{3L}{4}-1} Y_{i,j} \tag{3.1}$$

Algorithm 2 論文 [3] の目的画像に対する二値行列の生成

入力: サイズ $L \times L$ の目的画像の輝度 (Y) 成分 $Y_{i,j}(1 \leq i, j \leq L)$、閾値 $\overline{Y}$

出力: 二値行列 $B_{i,j}$

方法:
1. 目的画像を二値化する際、二値行列 $B_{i,j}$ は、以下の規則によって決定される．

$$B_{i,j} = \begin{cases} 1 & Y_{i,j} > \overline{Y} \\ 0 & otherwise \end{cases} \tag{3.2}$$

3.2 Random Method

Random Method とは最も見た目が目的画像に近い Aesthetic QR コードを生成する方法である. まず、AestheticQR コードの元となる QR コードを生成するための生成行列 G を求める. Random Method ではこの G を用いて、掃き出し法により QR コードを生成する. ただし、単位行列の列ベクトルの位置は情報記号の最初の $\hat{k}$ 列を除いてランダムに決定される. 次に、QR コードと目的画像の画像サイズを等しいものとして、QR コードの 1 モジュールと目的画像の 1 画素を対応させ、目的画像と QR コードがどれだけ近いものか比較する. ただし、QR コードに書き込むデータのシンボルを除いて比較する. 具体的には、目的画像の二値化を行ったものに QR コードのデータ (RS 符号中の情報記号の中のデータにあたる箇所) を書き加えたものを用意し、それと生成した QR コードとのハミング距離を取る．以上を N 回繰り返しハミング距離が最小となる QR コードを求め、それを使って Aesthetic QR コードを作成する．

本研究では QR コードの中でバージョン 1 の QR コードを用いた. バージョン 1 の QR コードを作成する手順を Algorithm3 に示す.

Algorithm 3 論文 [3] のランダム法を用いたバージョン 1 の Aesthetic QR コード

入力: バージョン 1 の QR コードに入るシンボル長 16 の文字列、サイズ 21×21 の目的画像

出力: サイズ 21×21 のバージョン 1 の Aesthetic QR コード

方法:

1. 目的画像の画素値を QR コードのモジュールに割り当て、Algorithm2 で決まった閾値 $\overline{Y}$ でモジュールを二値化し、二値行列 $B_{i,j}$ を作成する．
2. $B_{i,j}$ に所定のマスキングパターンを作用させる．
3. 式 (2.8) の α_t の位置 t ($\hat{k}+1 \leq t \leq n$) を変化させることにより、RS 符号を計算する．以後この手順を N 回繰り返すことにより、$B_{i,j}$ とのハミング距離が最小となる RS 符号を見つける．一定の試行回数終了後、QR コード上の RS 符号は、最良の RS 符号に置き換える．
4. マスク処理前である QR コードの各モジュールに対して、所定のマスクパターンを適用する．
5. 目的画像と QR コードをかさねて Aethetic QR コードを作成する.

第4章　数式処理システムMapleを用いたAestheticQRコードの実装

4.1　数式処理システム Maple

4.2　実験環境

実験に使用した PC 環境、言語を以下に示す.

- ソフトウェア実装環境
  - CPU：Intel(R)Core i5 7500 3.4GHz
  - OS：Windows 10 pro
  - 実装 RAM：16.0GB
- 開発環境
  - Maple：Maple 2021.1

実験に使用した各パラメータを以下に示す.

- QR コード
  - RS 符号：(26,16) 符号
  - 入力文字：tahara
  - バージョン (v)：1
  - マスクパターン：001
  - 誤り訂正レベル：M

QR コードのバージョン 1 の構造を図 4.1 に示す.

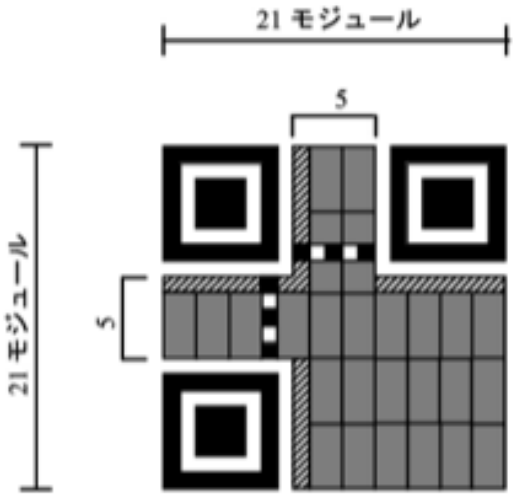

図 4.1　QR コードのバージョン 1 の構造

4.3　Maple による QR コードの実装

本研究では Aesthetic QR コードの生成に数式処理 Maple(以下、Maple) を用いた. 理由は RS 符号を生成する際に必要な有限体の実装を容易に行うことができるからである. 以下に Maple において有限体や各多項式をどのように用いて、QR コード中の RS 符号を生成したかを示す.

拡大体 $GF(2^8)$、原始多項式 $x^8+x^4+x^3+x^2+1$ とその原始元 α は以下のように表される. Maple において原始多項式は GF 関数の第 3 引数で定義され、原子元は a とした.

```
1 G8 := GF(2, 8, alpha^8+alpha^4+alpha^3+alpha^2+1):
2 a := G8:-ConvertIn(alpha):
```

```
> #関数名：gen_AestheticQRcode
  #入力１：QRコードの情報が入ったベクトル
  #入力２：目的画像を示すパス
  #出力：AestheticQRcodeの情報が格納された行列

  AestheticQRcode := gen_AestheticQRcode(AestheticQRcode_Vec,image_path):
  Preview(AestheticQRcode);
```

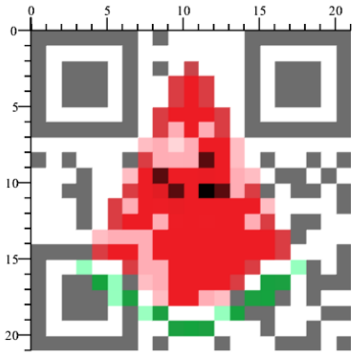

図 4.2　関数 AestheticQRcode の入力と出力

4.4　評価

Aestetic QR コードの評価は目的画像の二値行列 $B_{i,j}$ と得られた QR コードの二値行列における、符号化領域中のデータ及び誤り訂正コード語のハミング距離をとることで行う.

ハミング距離は Kuribayashi らの論文 [3] で使われている評価であり、小さければ RS 符号と目的画像の二値化行列が近いことを指す. RS 符号の生成にかかった時間は短いほど、手軽に作成することが可能であることがわかる. 時間は 10 回生成して平均をとったものを評価している.

4.4.1　最小ハミング距離

2 つの符号語つの符号語 $a = (a_1, a_2, \cdots, a_n)$ と $b = (b_1, b_2, \cdots, b_n)$ で対応するビット (桁）で値 (0 または 1) が異なっているビット (桁) の数をハミング距離と言い、記号で $d(a, b)$ と書く. その中でも一番ハミング距離が小さいものを最小ハミング距離と呼ぶ.

ハミング距離は 2 つの符号語 $a = (a_1, a_2, \cdots, a_n)$ と $b = (b_1, b_2, \cdots, b_n)$ に対して以下の式で定義される.

$$d(a,b) = \sum_{i=1}^{n} (a_i + b_i) \pmod 2 \tag{4.1}$$

4.5　実験結果

本実験では QR コードに挿入する画像は以下の目的画像 (図 4.3) を使用した. 図 4.4~4.8 はランダム法の試行回数が $N = 1, 10, 100, 1000, 10000$ の場合の結果 (Aestheic QR コード) を表す. 実際にソフトウェアによって生成した画像が図 4.4~ 図 4.8 である.

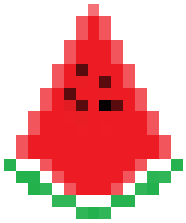

図 4.3　目的画像

図 4.4　$N = 1$

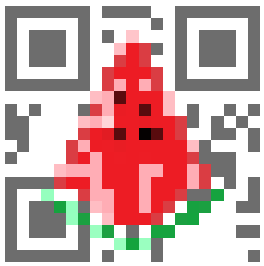

図 4.5　$N = 10$

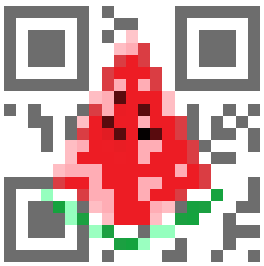

図 4.6　$N = 100$

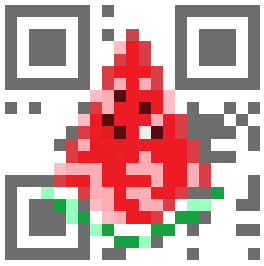

図 4.7　$N = 1000$

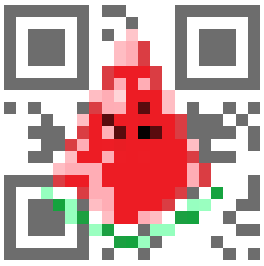

図 4.8　$N = 10000$

また、表 4.1 に各試行回数における実験で得られたハミング距離、RS 符号の生成にかかった時間を表す.

表 4.1　結果のまとめ

生成した回数 (N)	1	10	100	1000	10000
最小ハミング距離	83	63	51	47	41
時間（秒）	0.08	0.81	8.01	80.39	771.17

表 4.1 を見ると、試行回数を増やしていくたびに評価の値が向上しており、より良い Aesthetic QR コードが生成されているのがわかる. しかし、試行回数 10000 回の所を見ると RS 符号が生成するのにかかる時間が約 13 分もかかっており、計算時間の短縮が検討課題である.

第5章　結論

本研究では、数式処理システム Maple の CUI を利用し、Kuribayashi 論文の Random Method に則ってバージョン 1 の QR コードと目的画像から AestheticQR コードの実装を行った.

結果、AestheticQR コードの実装を行うことができ、その最小ハミング距離や生成時間の測定などを行うことができた.

本研究ではランダム法によって生成されるバージョン 1 の Aesthetic QR コードの Maple による実装を行った. 今後の課題として、

- 生成速度を短縮するアルゴリズムの実装
- 生成可能な Aesthetic QR コードのバージョンの追加

などが挙げられる.

謝辞

本研究に際して、日々、様々なご指導をいただきました甲斐博准教授に心より感謝致します. 最後に日頃から助言や励ましをいただきました諸先輩方、並びに同研究室の皆様に深く御礼を申し上げます.

参考文献

［1］ QR code.com. http://www.qrcode.com/en.

［2］ Visualead. http://www.visualead.com.

［3］ M. Kuribayashi and M . Morii ”Aesthetic QR Code Based on Modified Systematic Encoding Function”, IEICE transactions on information and systems ,VOL.E100–D, NO.1,pp.42-51,2017.

［4］ ユークリッド復号法, 技術の原点.https://www.jstage.jst.go.jp/article/essfr/4/3/4_3_183/_pdf

［5］ 池田和興, 例題が語る符号理論, 共立出版, 2007 年

［6］ J. Justesen and T. Hoholdt ”A Course In Error-Correctiong Codes”, European Mathematical Society Publishing House, 2004.

［7］ JIS X0510. 情報技術－自動認識及びデータ取得技術－ＱＲコード　バーコードシンボル体系仕様
http://www.jisc.go.jp/app/pager?id=2738494.

付 録A　プログラムリスト