ISSN 2186-5647

2-87

ジグソーパズルの模様を用いた情報秘匿の方法
－アンドロイド端末への実装－

日大生産工 (院)　○ 富樫 健二
日大生産工　伊藤 浩

1　まえがき

印刷された幾何学模様に情報を持たせる方法としてバーコードや QR コード [1] がある。しかし、それらのコードはデザイン性があまり考慮されていない。そこで、広く知れ渡っているジグソーパズルの模様を用いることによりデザイン性に富み、復号が容易な情報秘匿方法を提案する。復号にはフィルタや相関などの複雑な計算を必要としない。このパターンに印刷物を重畳すれば不正コピーの抑制や、書類の真がん判定にも応用が可能である [2]。

2　提案する方法

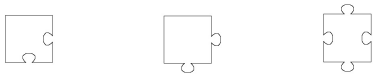
2.1　原理

ジグソーパズルのピース間の凹凸の曲線の部分に “0” か “1” の情報を持たせる。その区別はピースの内側に曲線が向くとき “0” とし、外側を向くときに “1” とする。各ピースは下側と右側の曲線に、この順に秘匿された 2bit の情報を持つ。図 1 にその例を示す。

本研究の情報伝達の経路は、まず情報を持たせた模様を印刷する。その印刷した模様をスキャナーかカメラで画像ファイルとして読み込み復号を行う。

2.2　符号方法

符号化方法は秘匿したい情報を左上から順にピースの下側と右側に秘匿したい情報に対応する曲線を下、右の順に順次書き足していく。この方法で生成した 24bit(110100010101011111000000) の情報を持ったジグソーパズルを図 2(a) に示す。



(a)pattern“00”　(b)pattern“11”　(c)pattern“10”

図 1: 基本パターン

2.3　復号方法

復号方法は以下の通りである。縦横のピース数はそれぞれわかっているものとする。始めに画像をモード法により 2 値化する。次に印刷物から模様のみを取り出し、スキャン時に生じる傾きを無くした後に縦横のピース数から各ピースの位置を見つける。そして、ピースの各曲線部分の辺の中点を基準として両側の一定範囲の黒画素数を比べ、どちら側の画素が多いかで各曲線の向きを調べる。最後に曲線の向きに対応する情報を出力する。

具体的な検出範囲の決定方法は、まず縦と横のピース数から各ピースの辺の位置を見つけ中点を探す。各ピースの中点は 1 つのピースの横と縦の長さをそれぞれ p、q とすると、横 i 個目、縦 j 個目のピースの曲線部分の中点の座標は下辺が $((i-1)\times p+p/2, j\times q)$、右辺は $(i\times p, (j-1)\times q+q/2)$ となる。2(b) にこれらの位置を示す。次に比較する領域の位置と大きさを決定する。ピースの大きさと曲線の大きさに比例関係を持たせているため、下辺では辺の中点から曲線の頂点までの距離 α を $\alpha_b = 0.2\times q$ とし、曲線の横幅 β を $\beta_b = 0.25\times p$ とする。右辺では $\alpha_r = 0.2\times p$ とし、$\beta_r = 0.25\times q$ とする。図 3(a) にその位置を示す。また、ピースの下側の領域 A と B, 右側の領域 A' と B' は曲線部分の中点から距離 $-\alpha$ と α 離れた各点を中心として長さ β と α の長方形の範囲をとる。図 3(b) に領域 A' と B' を示す。

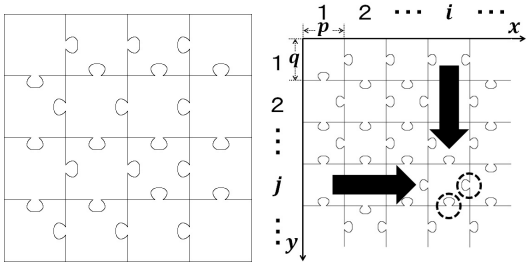
2.4　歪みの補正

Android 端末のカメラで撮影するときには三次元空間の奥行きにより撮影した模様に歪みが生じる。そこで歪みを取り除く方法として射影変換を使用する。射影変換は以下の式で表わされる。

$$\begin{pmatrix} sx' \\ sy' \\ s \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (1)$$

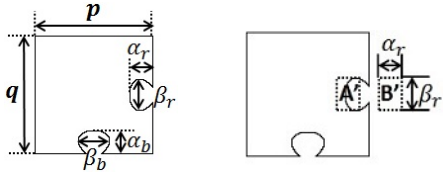
Information hiding in a Jigsaw puzzle pattern
－ Implementation in the Android device －

Kenji TOGASHI and Hiroshi ITO



(a) パズルの例　(b) 検出位置

図 2: ジグソーパズルの模様



(a) 検出位置の詳細　(b) 比べる領域

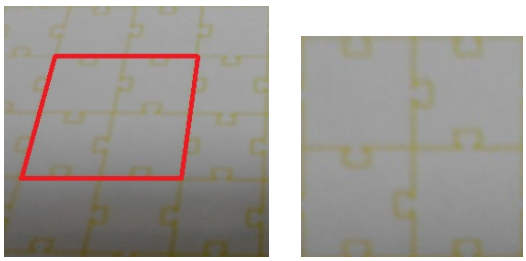
図 3: 模様の検出

式 (1) は式 (2) のように変換できる。

$$x' = \frac{ax+by+c}{gx+hy+1}, y' = \frac{dx+ey+f}{gx+hy+1} \qquad (2)$$

この式は画像の回転、移動を行う式である。ワールド座標とカメラ座標の座標軸の関係が分かれば、カメラ座標からワールド座標を求めることが可能である。式 (1) の a〜h がその関係を表すパラメータである。つまり a〜h の 8 個のパラメータが求まれば射影変換が可能となる。具体的には変換前 (x_1〜x_4, y_1〜y_4) と変換後 (x'_1〜x'_4, y'_1〜y'_4) の対応点である 4 組の点の座標を用いて、式 (2) の 2 つの式にそれぞれ座標を代入し 8 個の連立方程式からパラメータを求める。パラメータが求められたら式 (1) に代入して各座標の変換を行う。

射影変換を利用すると図 4(a) に示す線に囲われている領域が図 4(b) のように正面から撮影した画像に変換される。

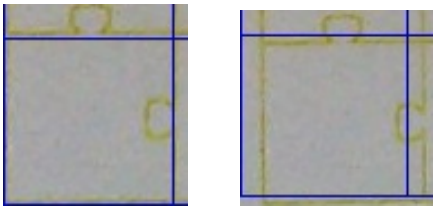


(a) 変換前　(b) 変換後

図 4: 射影変換

3　Android アプリの開発

開発環境は Eclipse、言語は Java で Android SDK を利用してアプリケーションの開発を行った。アプリケーションは大まかに分けて 2 種類の動作を行う。1 つはカメラを起動して写真を撮ること。もう 1 つは撮影した写真から模様を取り出して復号を行い、結果を表示することである。アプリケーションの使用方法は、まず Android 端末のカメラを使用し模様を撮影する。その後、画面が切り替わり撮影した写真内から射影変換を用いて模様のみを取り出して、この画像を復号結果と共に画面に表示をする。



(a) 変換有り　(b) 変換無し

図 5: 射影変換の有無

4　検証実験

カメラで撮影した模様に対して射影変換を実行しない場合と実行する場合を比較する。使用する Android 端末は Nexus7 である。カメラの画素数は 500 万画素である。なお、模様を撮影する際にはオートフォーカスを使用する。撮影する模様は大きさは 7.8cm、ピースは 8×8、情報量は 112bit である。カメラは模様から約 30cm 離して撮影をした。画像内の模様の横のピクセル数と印刷した模様の大きさから解像度は約 194dpi であった。射影変換を実行した場合と実行しない場合の撮影した模様の左下の隅を図 5 にそれぞれ示す。図の青い線が復号プログラムが画像の大きさと縦横のピース数によって求めたピースの位置である。この位置から図 3 の比較する領域が決定される。

射影変換を実行した後に復号をすると誤り率は 0%であったが、射影変換を実行しないで復号すると誤りが 5 箇所になり誤り率は 4.46%になった。

5　まとめ

ジグソーパズルの模様を用いた情報秘匿の方法を提案した。射影変換をすることによって誤りが少なくなることが分かる。つまり、カメラで撮影した場合でも画像内に生じた歪みは射影変換を実行することによって補正できるため、提案した方法は Android 端末で使用が可能である。また、射影変換の有無で誤り率に違いがでた原因としては射影変換を行わないと復号の際にピースの位置がプログラムの想定した位置から大幅にずれることにより図 3 の比較領域もずれてしまうためだと考えられる。今後は復号の際に結果が誤ってしまっても復号時に誤りの訂正が可能となるように、誤り訂正符号を導入する。

参考文献

[1] 長屋, 他:高速読取り対応 2 次元コード [QR コード] の開発, 情報処理全国大会 (1996)

[2] 菅井, 他:改ざん検出可能な印刷物対応電子透かしの開発, 電子情報通信学会総合大会 (2009)