Le tatouage de documents numérique
Cours 3

Marc Chaumont

18 novembre 2008

Plan

Schéma générique

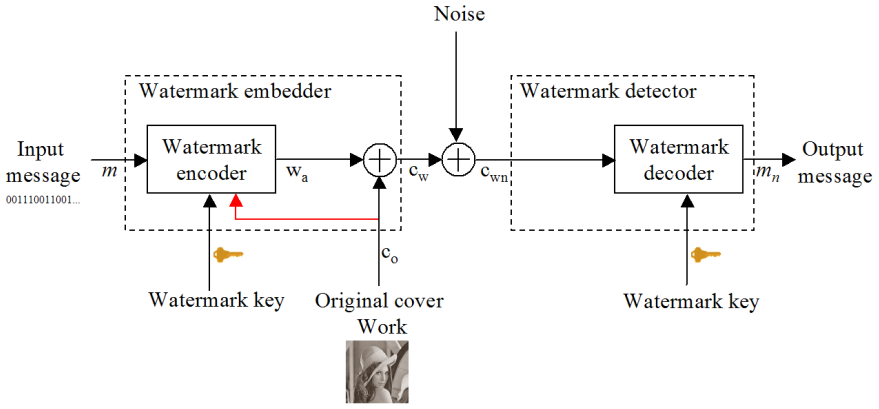
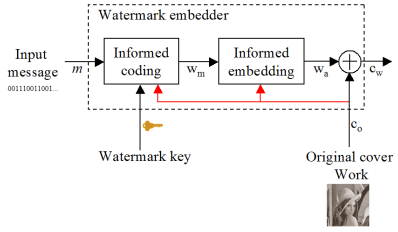


Schéma plus précis



On peut distinguer (dans certains cas) :

- étape 1 : **le codage informé** (codage du message) (cf. résultat de Costa 83),
- étape 2 : **l'insertion informée** (l'insertion à proprement parler du message codé dans le signal hôte).

Plan

Énoncé du problème (optimisation)

Le problème de l'insertion peut être vu comme un problème d'optimisation :

- Soit trouver la marque qui maximise la robustesse (position dans l'hyper-plan ou l'hyper-cône) tout en conservant une distortion perceptuelle fixée (distance de Watson ...) (cas non détaillé dans ce cours).
- Soit trouver la marque qui minimise la distortion perceptuelle tout en maintenant un niveau de robustesse fixé (cas non détaillé dans ce cours)

Mesure - définition - de la robustesse

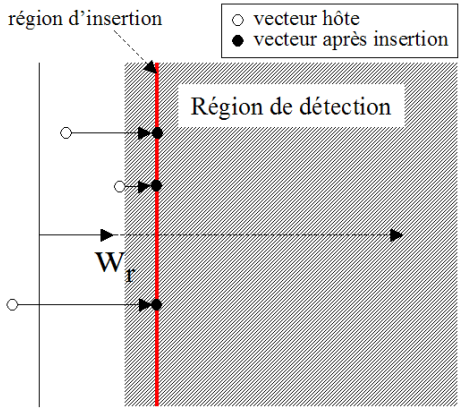
Une **mesure simpliste de robustesse** est de supposer qu'une marque insérée avec une valeur de corrélation forte est plus robuste qu'une marque insérée avec une faible valeur de corrélation.

- c'est **vrai** pour la corrélation linéaire,
- c'est **faux** pour la corrélation normalisée.

Les transparents suivants expliquent ce point.

Plan

Insertion informée avec une stratégie d'insertion à corrélation linéaire fixée



Modélisation de la stratégie d'insertion à corrélation linéaire fixé

La corrélation linéaire entre un signal marqué $c_w = c_o + \alpha w_m$ et un pattern w_m de taille N est :

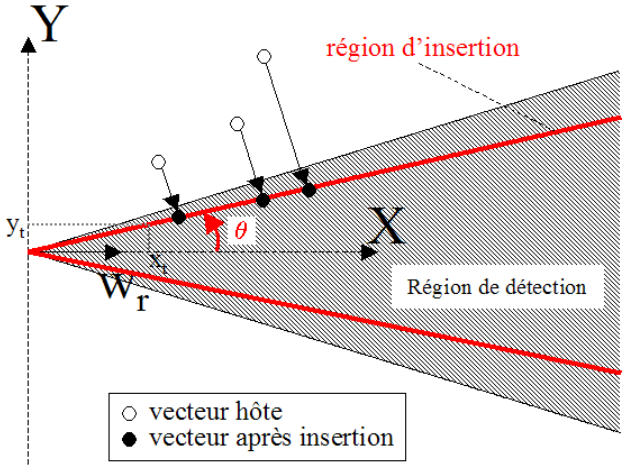
$$z_{lc}(c_w, w_m) = \frac{1}{N}(c_o.w_m + \alpha w_m.w_m)$$

Si le seuil de détection est fixé à $\tau_{lc} + \beta$, on peut déduire la valeur de α :

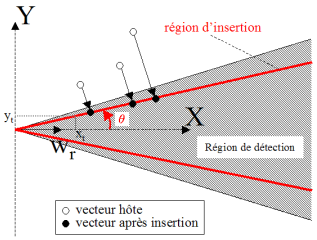
$$\alpha = \frac{N(\tau_{lc} + \beta) - c_o.w_m}{w_m.w_m}$$

Plan

Insertion informée avec une stratégie d'insertion à corrélation normalisée fixée



Insertion informée avec une stratégie d'insertion à corrélation normalisée fixée



Solution :

- fixer la corrélation nomalisée à τ_{nc}
- trouver le point c_w issu de la projection du signal hôte c_o sur la surface du cône. Le cône est centré sur le pattern de référence w_r.

Note : c_o, c_w et w_r appartiennent au même plan.

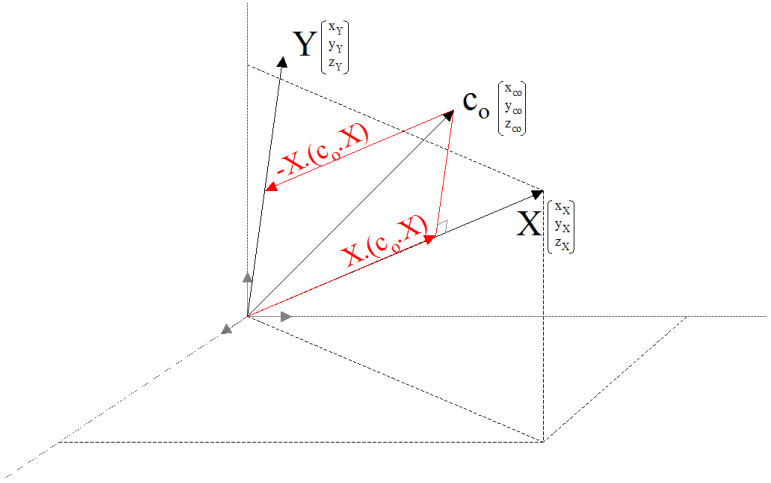
Réduction du problème à un problème à 2 dimension : définition du "plan de Cox"

Soit (X,Y) deux axes orthogonaux définissant le plan contenant c_o et w_r. On utilise la technique d'orthonormalisation de **Gram-Schmidt** :

$$X = \frac{w_r}{|w_r|}$$

$$Y = \frac{c_o - X.(c_o.X)}{|c_o - X.(c_o.X)|}$$

Illustration de l'orthonormalisation de Gram-Schmidt pour une image c_o à 3 pixels (3 dimensions)

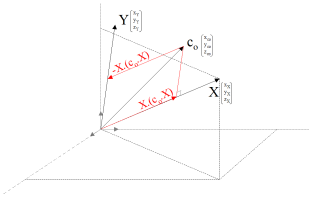


(1) Code pour le calcul de la région d'insertion

```
/*-----------------------------------------------------------------------*
 | Orthonormalize -- convert two vectors into two unit-length, orthogonal  |
 |                   vectors that lie in the same plane                    |
 | Arguments:                                                              |
 |   X -- vector who's direction will not be changed (changed in place)    |
 |   Y -- vector who's direction will be changed (changed in place)        |
 |   size -- number of component for a vector                              |
 *-----------------------------------------------------------------------*/
void WMTools::Orthonormalize( double *X, double *Y , int size) {
  double XDotY;                       /* inner product of original Y and
                                         unit-length X */
  double len;                         /* Euclidian length (magnitude) of a
                                         vector */
  int i;                              /* index into marks */

  /* Normalize X to unit length. */
  len = 0;
  for( i = 0; i < size; i = i + 1 )
    len = len + X[ i ] * X[ i ];
  len = sqrt( len );
  for( i = 0; i < size; i = i + 1 )
    X[ i ] = X[ i ] / len;

  ...
```

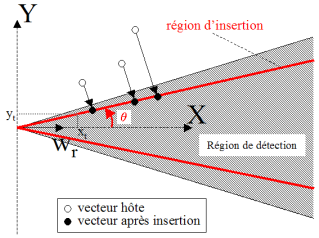
(2) Code pour le calcul de la région d'insertion

```
  ...

  /* Subtract X * (X dot Y) from Y to ensure that X and Y are orthogonal. */
  XDotY = PatPatInnerProduct( X, Y , size);
  for( i = 0; i < size; i = i + 1 )
    Y[ i ] = Y[ i ] - XDotY * X[ i ];

  /* Normalize Y to unit length. */
  len = 0;
  for( i = 0; i < size; i = i + 1 )
    len = len + Y[ i ] * Y[ i ];
  len = sqrt( len );
  for( i = 0; i < size; i = i + 1 )
    Y[ i ] = Y[ i ] / len;
}
```

Chaque point du plan (**X**,**Y**) peut être exprimé par un point 2D (x,y) et dans le repère intiale par $x\mathbf{X} + y\mathbf{Y}$.

Equation de la région d'insertion

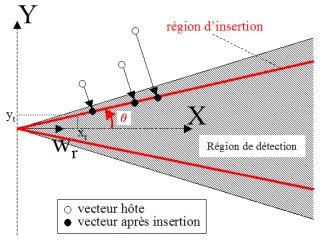


La région d'insertion est un cône et donc une droite $\mathcal{D}$ dans le plan (**X**,**Y**). La droite $\mathcal{D}$ peut être décrite par un vecteur (x_t, y_t) dans le plan (**X**,**Y**) :

$$\begin{aligned} x_t &= cos(\theta) \\ y_t &= sin(\theta) \end{aligned}$$

avec θ l'angle entre w_r et la droite $\mathcal{D}$.

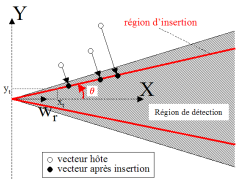
Équation de la région d'insertion



$\theta = cos^{-1}(\tau_{nc})$ avec τ_{nc} le seuil de détection (corr. norm.) fixé par l'utilisateur. D'où :

$$\begin{aligned} x_t &= \tau_{nc} \\ y_t &= sin(cos^{-1}(\tau_{nc})) = \sqrt{1-\tau_{nc}^2} \end{aligned}$$

Équation de la région d'insertion



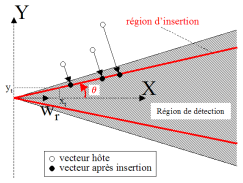
Les coordonnées de c_o dans le plan (X,Y) sont

$$\begin{aligned} x_{c_o} &= c_o.X \\ y_{c_o} &= c_o.Y \end{aligned}$$

Le point $c_w = (x_{c_w}, y_{c_w})$ (exprimé dans le plan (X,Y)) le plus proche (distance euclidienne) de c_o appartenant à $\mathcal{D}$ est le point :

$$\begin{aligned} x_{c_w} &= x_t.(x_t.x_{c_o} + y_t.y_{c_o}) \\ y_{c_w} &= y_t.(x_t.x_{c_o} + y_t.y_{c_o}) \end{aligned}$$

Équation de la région d'insertion



L'équation du point c_w exprimé dans le repère initial est :

$$c_w = x_{c_w}.X + y_{c_w}.Y$$

Code pour le calcul de la région d'insertion

```
/*-----------------------------------------------------------------------*
 | MixFixedCC -- compute a vector that is close to a given extracted     |
 |               vector, and has a fixed correlation coefficient with a  |
 |               given message mark                                      |
 |                                                                       |
 | The correlation between the new vector and the reference vector is    |
 | specified as the sum of a detection threshold and a "strength"        |
 | parameter.  The new vector is as close as possible to the given       |
 | extracted vector, measured by Euclidian distance.                     |
 |                                                                       |
 | Arguments:                                                            |
 |   co -- original image                                                |
 |   tcc -- detection threshold                                          |
 |   beta -- strength parameter                                          |
 |   wr -- message mark                                                  |
 |   vw -- where to store resulting vector                               |
 *-----------------------------------------------------------------------*/

void WME_FIXED_CC::MixFixedCC( unsigned char* co, int width, int height,
                               double tcc, double beta, double *wr, double *cw ) {

  double* X = new double [ width*height ]; /* unit vector aligned with wm */
  double* Y = new double [ width*height ]; /* unit vector orthogonal to X, such
                                              that X and Y describe the plane
                                              containing wm, co and the origin */

  ...
```


Code pour le calcul de la région d'insertion

```
...

double wrMean;                          /* mean of wr */
double coMean;                          /* mean of co */
double xco, yco;                        /* coordinates of co in the XY plane */
double xt, yt;                          /* unit vector in the XY plane that
                                           has the desired correlation with
                                           the watermark */
double xcw, ycw;                        /* coordinates of new vector in the
                                           XY plane */
int i;                                  /* index into vectors */

/* An initial version of X. */
wrMean = WMTools::MarkMean( wr , width*height );
for( i = 0; i < width*height; i = i + 1 )
  X[ i ] = wr[ i ] - wrMean;

/* An initial version of Y. */
coMean = WMTools::MarkMean( co , width*height );
for( i = 0; i < width*height; i = i + 1 )
  Y[ i ] = (double) co[ i ] - coMean;

/* Apply Gram-Schmidt orthonormalization to obtain two orthogonal
   unit vectors. */
WMTools::Orthonormalize( X, Y , width*height);

...
```

Code pour le calcul de la région d'insertion

```
...

/* Find projection of co into the XY plane. */
xco = WMTools::ImgPatInnerProduct( co, X, width, height );
yco = WMTools::ImgPatInnerProduct( co, Y , width, height );

/* Find unit vector in the XY plane that has a normalized correlation
   with the watermark of tcc + beta */
xt = tcc + beta;
yt = sqrt( 1 - xt * xt );

/* Find the point on the line described by xt,yt that is closest to
   xco,yco */
xcw = xt * (xt * xco + yt * yco);
ycw = yt * (xt * xco + yt * yco);

/* Project xcw,ycw back into mark space */
for( i = 0; i < width*height; i = i + 1 )
  cw[ i ] = xcw * X[ i ] + ycw * Y[ i ] + coMean;

/* Delete X and Y */
delete [] X;
delete [] Y;
}
```


Réflexion sur la robustesse des deux approches

- Pour la correlation linéaire, augmenter le seuil de détection revient à augmenter la robustesse (on déplace l'hyper-plan),
- Pour la corrélation normalisée, augmenter le seuil de détection revient à diminuer l'angle du cône et donc obtenir un point c_w plus proche de l'origine du cône. Un bruit (attaque) sur l'image c_w peut alors plus facilement faire sortir le point du cône de détection.

Plan

Réflexion sur la robustesse de la corrélation normalisée

La corrélation normalisée ne mesure pas directement la robustesse.

Il faut trouver une mesure fonction d'un bruit que peut subir le vecteur c_w avant de sortir de la région de détection. Il est interessant de choisir un bruit blanc Gaussian car la corrélation normalisée n'y est pas spécialement robuste et de plus c'est un modèle simple.

Équation de la robustesse pour un niveau de bruit fixé

Supposons un bruit blanc Gaussien **n** ajouté à l'image tatouée c_w modélisant une attaque. La corrélation normalisée au détecteur est alors :

$$z_{nc}(c_w + n) = \frac{(c_w + n).w_r}{|c_w + n||w_r|}$$

En supposant que le bruit **n** est probablement orthogonal à c_w et à w_r, nous obtenons :

$$z_{nc}(c_w + n) \approx \frac{c_w.w_r}{\sqrt{c_w.c_w + n.n}|w_r|}$$

Équation de la robustesse pour un niveau de bruit fixé

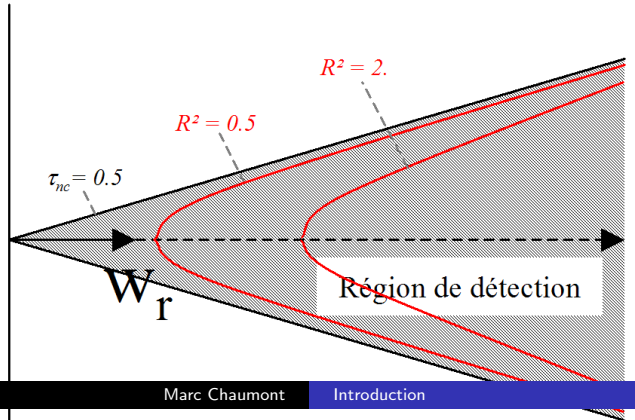
Nous cherchons à trouver l'amplitude du bruit $R = \sqrt{n.n}$ qui fait passer $z_{nc}(c_w + n)$ sous le seuil τ_{nc}. En remplacant $z_{nc}(c_w + n)$ par τ_{nc}, nous obtenons :

$$R^2 = (\frac{c_w.w_r}{\tau_{nc}|w_r|})^2 - c_w.c_w \quad (1)$$

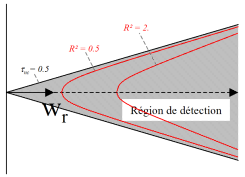
... si l'on fixe un niveau de bruit R^2, on peut déduire c_w. On vient donc de définir une **mesure grossière** de robustesse.

Représentation graphique de la région d'insertion pour un niveau de bruit R^2 constant

Pour être robuste à un bruit **n** blanc Gaussien $R^2 = n.n$ constant, fixé par l'utilisateur, la zone d'insertion (zone de c_w) est une moitié d'hyperbole de dimension N.



Solution approchée de la projection sur la demi hyperbole

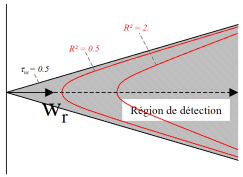


$$R^2 = (\frac{c_w.w_r}{\tau_{nc}|w_r|})^2 - c_w.c_w$$

À l'insertion, on doit résoudre une équation quadratique pour déterminer c_w sachant co, w_r et R^2. Par simplicité, on peut faire une recherche exhaustive en testant la distance entre c_o (exprimée dans (**X**,**Y**)) et des points d'abscisses dans l'intervalle $[0, y_{c_o}]$ appartenant à l'hyperbole.

Solution approchée de la projection sur la demi hyperbole

$$R^2 = (\frac{c_w.w_r}{\tau_{nc}|w_r|})^2 - c_w.c_w$$



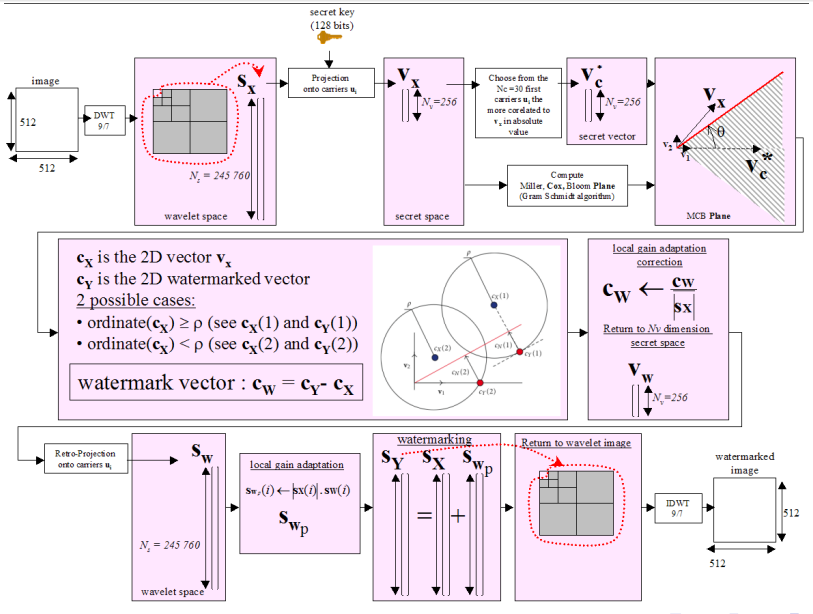
Dans le repère (**X**,**Y**), soit $c_w = (x_{c_w}, y_{c_w})$ et $w_r = (|w_r|, 0)$. On a alors :

$$x_{c_w} = \frac{\tau_{nc}^2(R^2 + y_{c_w})}{1 - \tau_{nc}^2}$$

Plan

Insertion par "flèche brisée"

- "**Broken Arrows**", Teddy Furon and Patrick Bas, EURASIP Journal on Information Security, Volume 2008 (2008), Article ID 597040, 13 pages, doi :10.1155/2008/597040, 21 August 2008.
- Schéma de tatouage (0-bit) utilisé lors de la compétition BOWS-2 (http ://bows2.gipsa-lab.inpg.fr/)
- L'insertion par l'approche *flèche brisée* est plus robuste que celle par *hyper-hyperbole*.



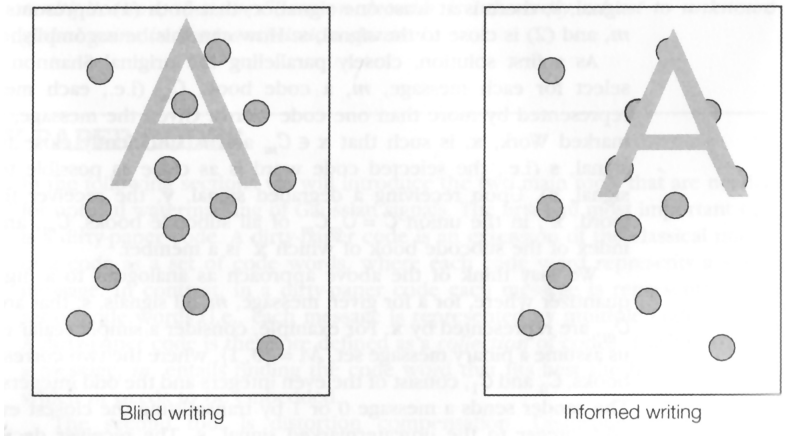
Plan

Résultat de Costa 1983

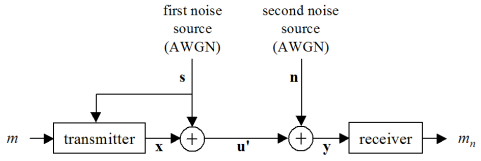
Problème :

Soit un papier couvert de "taches de saleté" d'intensités distribuées selon une loi normale. On écrit alors un message sur ce papier avec une quantité limitée d'encre. Le papier sale est alors envoyé avec le message et est également taché selon une loi normale. Si le récepteur ne peut distinguer l'encre de la saleté, combien d'information fiable peut-on envoyer ?

Ecrire sur un papier sale



Résultat de Costa 1983



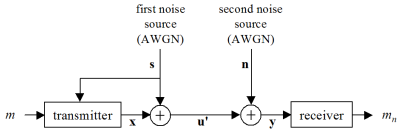
Problème équivalent :

Un canal de transmission possède 2 sources de bruit indépendantes Gaussienne. Avant de transmettre x, l'émetteur connait s le premier bruit. Le signal x est transmis avec une puissance limitée : $\frac{1}{N}\sum_i x[i] \leq p$. Le deuxième bruit n est inconnu.

Résultat prouvé par Max Costa en 1983

Le premier bruit s n'a pas d'influence sur la capacité du canal

Résultat de Costa 1983



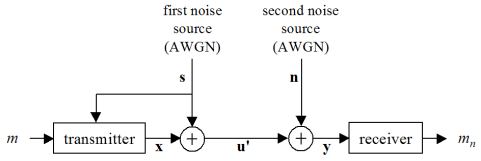
Résultat prouvé par Max Costa en 1983

Le premier bruit s n'a pas d'influence sur la capacité du canal

Capacité de Shannon-Hartley (en bits)	Capacité de Costa (en bits)
$C_{noSI} = B.log_2(1 + \frac{\sigma_x^2}{\sigma_s^2+\sigma_n^2})$	$C_{SI} = B.log_2(1 + \frac{\sigma_x^2}{\sigma_n^2})$

Résultat de Costa 1983 : Illustration

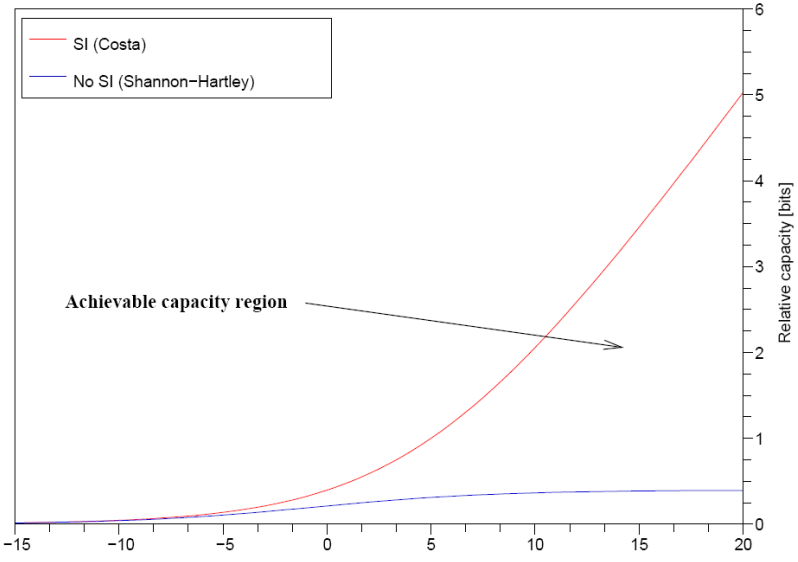
$w \sim \mathcal{N}(0, \sigma_w^2)$, $x \sim \mathcal{N}(0, \sigma_x^2)$, $n \sim \mathcal{N}(0, \sigma_n^2)$.



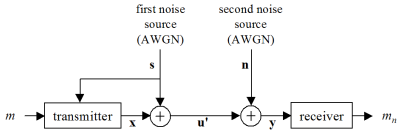
On fixe $\sigma_x^2 \approx 0.316$, $\sigma_s^2 = 1$ (ce qui correspond à un $WCR = 10.log_{10}(\frac{\sigma_x^2}{\sigma_s^2}) = -5$ dB) et B = 1.

On va faire varier le bruit n c'est à dire la variance σ_n^2 c'est à dire le $WNR = 10.log_{10}(\frac{\sigma_x^2}{\sigma_n^2})$ et observer la capacité C_{SI} et C_{noSI}.

Résultat de Costa 1983 : Illustration



Résultat de Costa 1983

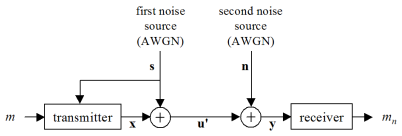


Ce résultat peut être appliqué pour le tatouage :

- s (le papier) $\equiv$ signal hôte,
- x (l'écriture du message) $\equiv$ pattern ajouté,
- p (quantité d'encre) $\equiv$ contrainte de fidelité,
- n (saleté supplémentaire) $\equiv$ attaque.

Le résultat de Costa, appliqué au tatouage indique que la quantité d'information que l'on peut embarquer via le tatouage ne dépend pas du signal couverture. La restriction quant à la quantité d'information que l'on peut insérer est due au deuxième bruit ; une redondance de codage est donc nécessaire pour que la tranmission soit correcte.

Résultat de Costa 1983 : remarque



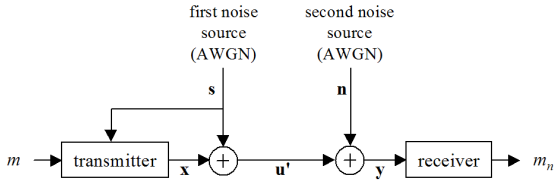
- Le résultat de Costa n'est vrai que dans le cas d'un codage avec utilisation d'un dictionnaire de mot-de-code choisi aléatoirement.
- Les deux bruits sont des bruits gaussiens blancs.
- Même si les hypothèses sont irréalistes dans le cas du tatouage, le résultat de Costa a mis en évidence :
  - l'apport de l'information adjacente (augmentation des capacités et amoindrissement de l'impact du bruit support),
  - et également donné des pistes pour la contruction des codes.

1998-... Tatouage informé

En 1998, le résultat de Costa est **re-découvert**, transposé au tatouage et les premières solutions de tatouage informé ont alors été proposées.

En référence au papier de Costa, nous parlerons de **"dirty-paper code"** pour la classe de code utilisée pour le codage de message pour le **tatouage avec information adjacente**.

Tatouage informé : Principe



On souhaite **trouver un mot de code** u' représentant (codant) le message m tel que le mot de code u' soit **proche de** s. La distance maximum entre u' et s est de $\sqrt{Np}$ (du fait de la contrainte $\frac{1}{N}\sum_i^N x^2[i] \leq p$ cad $\sqrt{\sum_i^N x^2[i]} \leq \sqrt{Np}$).

Tatouage informé : Principe

Par exemple, on souhaite transmettre un des 4 messages A,B,C,D. On choisit un ensemble $\mathcal{U}$ de vecteurs, on le divise en 4 sous-ensemble (coset) $\mathcal{U}_A$, $\mathcal{U}_B$, $\mathcal{U}_C$, $\mathcal{U}_D$. Chaque mot de code d'un coset représente un message.

Le codeur choisit lorsqu'il veut coder un message m le mot de code u du (coset $\mathcal{U}_m$) qui est le plus proche du vecteur hôte s.

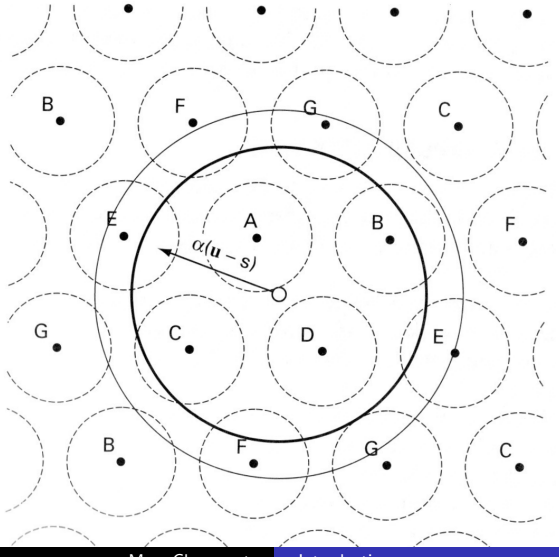
Le codeur transmet alors $x = u - s$ (cf. schéma).

Le récepteur reçoit le vecteur y et détermine le mot de code u'' le plus proche de y, identifie le coset $\mathcal{U}_{m'}$ et donc le message m'.

Tatouage informé : Principe

Problème de l'approche : il est possible que certains mots de code u représentant un message m soient trop éloignés de s (contrainte de puissance non respectée). Plutôt que de transmettre $x = u - s$ on préfère transmettre $x = \alpha(u - s)$ avec α un scalaire de $[0, 1]$. Remarque : après l'ajout du signal hôte, on a $u' = \alpha u + (1 - \alpha)s$. Le choix de α dépend du canal.

Tatouage informé : Principe



Plan

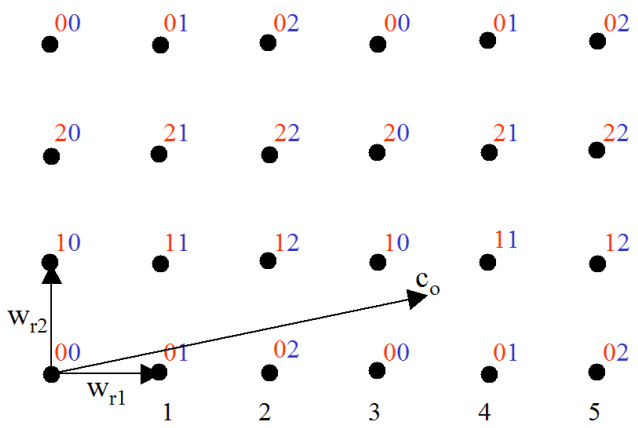
Information adjacente avec les codes à lattice : DIM (Dither Index Modulation)

Le point (mot de code) le plus proche de c_o dans la sous-lattice représentant un message $m = m[1], m[2], ..., m[K]$ est le vecteur z_m dont les composantes i sont données par :

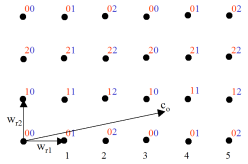
$$z_m[i] = a\lfloor \frac{l[i]/|w_{ri}| - m[i]}{a} + 0.5 \rfloor + m[i]$$

- $a = |\mathcal{A}|$ taille de l'alphabet des symboles,
- w_{ri} **i-ème pattern (porteuse)** de taille N (taille de c_o),
- $l[i] = c_o.w_r/|w_{ri}|$ la longueur de la projection de c_o sur w_{ri}.

Lattice à dictionnaire à 3 symboles



Exemple de codage



Le vecteur c_o projeté sur w_{r1} donne $l[1]/|w_{r1}| = 3.4$. On souhaite déterminer la i-ème composante $z_m[i]$ (du mot de code z_m) codant $m[1] = 2$.

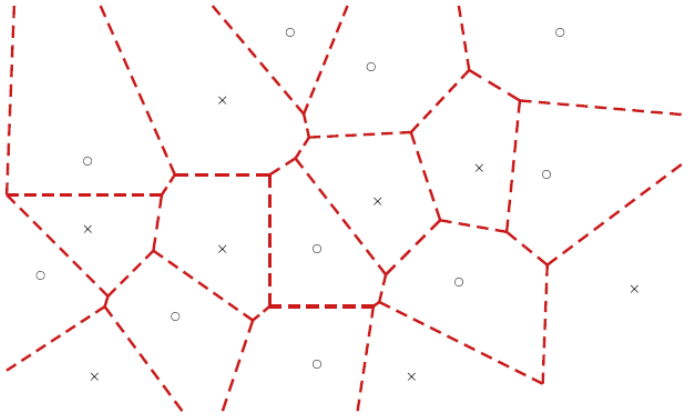
$$\begin{aligned} z_m[i] &= a\lfloor \frac{l[i]/|w_{ri}| - m[i]}{a} + 0.5 \rfloor + m[i] \\ &= 3\lfloor \frac{3.4 - 2}{3} + 0.5 \rfloor + 2 \\ &= 3\lfloor \frac{1.4}{3} + 0.5 \rfloor + 2 = 3 \times 0 + 2 = 2 \end{aligned}$$

Technique de tatouage QIM

Soit un signal hôte c_o de dimension N.

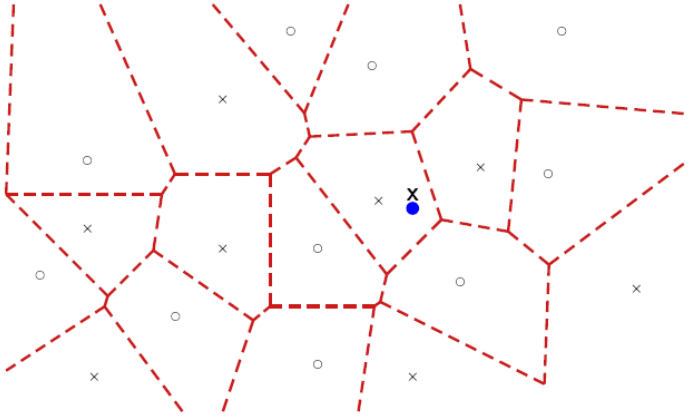
- déterminer les K entiers $z_m[i]$ (equation précedente) qui décrivent le mot de code le plus proche dans la sous-lattice pour un message m donné,
- la marque-message est alors $w_m = \sum_{i=1}^{K} z_m[i].w_{ri}$
- appliquer une insertion informée pour obtenir la marque à ajouter au signal hôte c_o. Par exemple $w_a = \alpha(w_m - c_o)$

A Voronoï view on QIM and SCS



Binary (DC-)QIM. Quantizer for 0's (resp. 1's) are the ○'s (resp. ×'s).

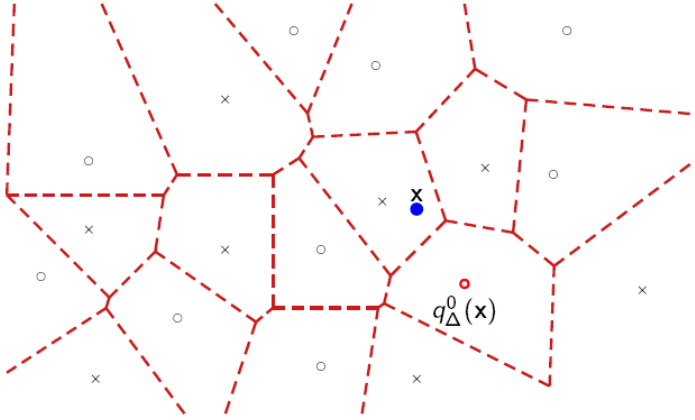
A Voronoï view on QIM and SCS



Binary (DC-)QIM. Quantizer for 0's (resp. 1's) are the ○'s (resp. ×'s).

Tatouage informé : 1998 - ...
Etape 2 : L'insertion
Etape 1 : Dirty paper codes
Sujet stage 2008-2009 + Examen
Introduction
Codes à "lattice" (en francais : grille - réseau)
Dirty-paper trellis code
Autres dirty-paper codes

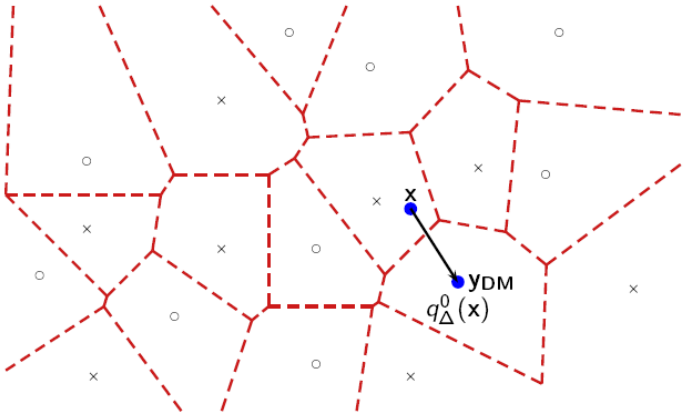
A Voronoï view on QIM and SCS



Binary (DC-)QIM. Quantizer for 0's (resp. 1's) are the ○'s (resp. ×'s).

Tatouage informé : 1998 - ...
Etape 2 : L'insertion
Etape 1 : Dirty paper codes
Sujet stage 2008-2009 + Examen
Introduction
Codes à "lattice" (en francais : grille - réseau)
Dirty-paper trellis code
Autres dirty-paper codes

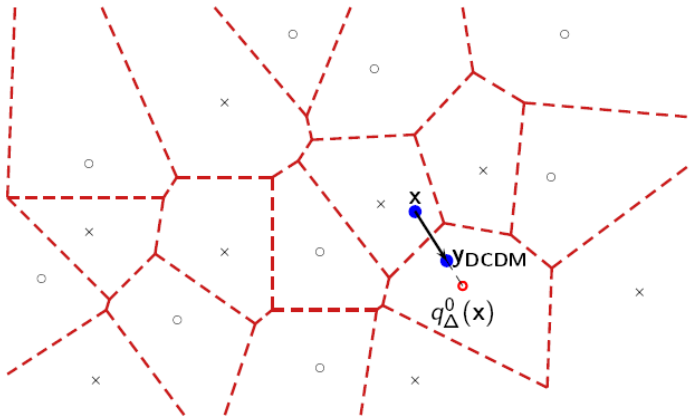
A Voronoï view on QIM and SCS



Binary (DC-)QIM. Quantizer for 0's (resp. 1's) are the ○'s (resp. ×'s).

Tatouage informé : 1998 - ...
Etape 2 : L'insertion
Etape 1 : Dirty paper codes
Sujet stage 2008-2009 + Examen
Introduction
Codes à "lattice" (en francais : grille - réseau)
Dirty-paper trellis code
Autres dirty-paper codes

A Voronoï view on QIM and SCS



Binary (DC-)QIM. Quantizer for 0's (resp. 1's) are the ○'s (resp. ×'s).

Exemple de schéma complet basé sur QIM - Codeur

Le système embarque 1 bit pour 256 pixels. On dispose d'une marque w_r de taille 8×8.

1. Pour plus de robustesse, nous codons le message binaire par code correcteur convolutif. Exemple : 345 bits sont codés en 1380 bits pour une image 240×368. On dispose alors d'une séquence de bits $m[1], m[2], ..., m[1380]$,
2. ...
3. ...

Deux mots sur le codage correcteur d'erreur par code convolutif

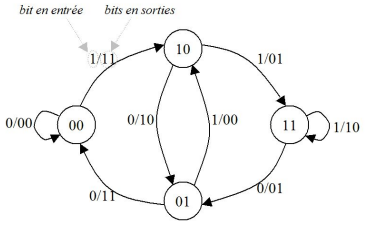


diagramme d'états-transitions

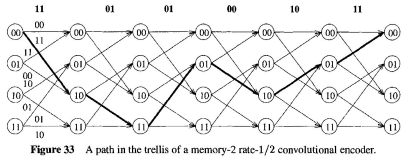


Figure 33 A path in the trellis of a memory-2 rate-1/2 convolutional encoder.

codage de 110100

Exemple de schéma complet basé sur QIM - Codeur

Le système embarque 1 bit pour 256 pixels. On dispose d'une marque w_r de taille 8×8.

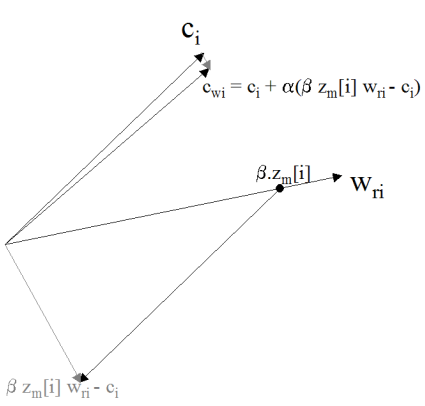
1. Pour plus de robustesse, nous codons le message binaire par code correcteur convolutif. Exemple : 345 bits sont codés en 1380 bits pour une image 240×368. On dispose alors d'une séquence de bits $m[1], m[2], ..., m[1380]$,
2. Diviser l'image c_o en (1380) blocs de taille 8×8,
3. Modifier chaque bloc pour insérer 1 bit. Soit c_i le ième bloc :

$$
\begin{aligned}
l[i] &= \frac{c_i . w_r}{|w_r|} \\
z_m[i] &= 2\lfloor \frac{l[i]/(\beta|w_r|) - m[i]}{2} + 0.5 \rfloor + m[i] \\
w_{ai} &= \alpha(\beta z_m[i] w_r - c_i) \\
c_{wi} &= c_i + w_{ai}
\end{aligned}
$$

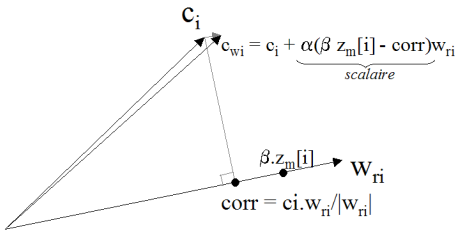
On peut prendre par exemple $\alpha = 0.9$ et $\beta = 3$

Illustration de l'insertion

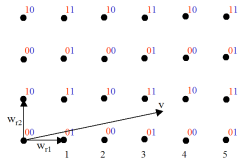
solution proposée



solution implémentée



Exemple de schéma complet basé sur QIM - Décodeur



Le détecteur est très simple :

1. Diviser l'image c_{wn} en (1380) blocs de taille 8×8.
2. Pour chaque bloc $c_{wn,i}$ calculer $z[i]$ (quantification du signal la lattice) :

$$z[i] = \lfloor \frac{c_{wn,i}.w_r}{(\beta w_r.w_r)} + 0.5 \rfloor$$

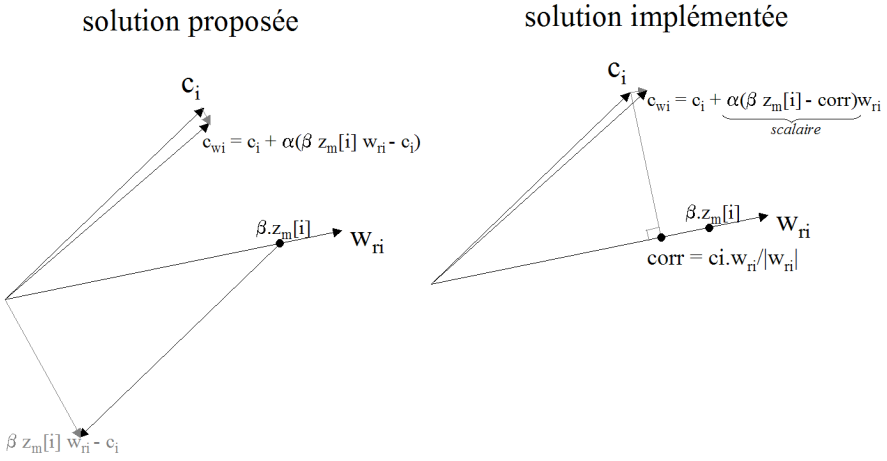
Exemple de schéma complet basé sur QIM - Décodeur

1. Pour chaque bloc $c_{wn,i}$ le bit extrait est :

$$m_c[i] = \begin{cases} 0 \ si \ z[i] \ est \ pair \\ 1 \ si \ z[i] \ est \ impair \end{cases}$$

2. une fois que le vecteur m_c est extrait (1380 bits), on décode par décodage convolutif (algorithme de viterbi) pour obtenir le message m (345 bits).

Illusration de l'insertion



Exemple de schéma complet basé sur QIM

```
/*-----------------------------------------------------------------------*
 | E_LATTICE -- embed a message using lattice-coded watermarking         |
 |                                                                       |
 | Arguments:                                                            |
 |   c -- image in which to embed (changed in place)                     |
 |   width -- width of image                                             |
 |   height -- height of image                                           |
 |   m -- message to embed (array of integer bit values)                 |
 |   mLen -- length of message (number of bits)                          |
 |   beta0 -- lattice spacing (as a multiple of the length of wr0)       |
 |   alpha  -- weighting factor between mark and cover signal            |
 |   wr0 -- basic reference mark (8x8 pattern)                           |
 |                                                                       |
 | Return value:                                                         |
 |   none                                                                |
 *-----------------------------------------------------------------------*/

void WM_LATTICE::E_LATTICE( unsigned char *c, int width, int height,
                            int *m, int mLen, double beta0,
                            double alpha, double *wr0 )

  ...
```

Exemple de schéma complet basé sur QIM

```
beta;                               /* value of beta after normalizing */
double wr[ 64 ];                    /* value of wr0 after normalizing */
int* mc = new int [ width * height]; /* message after coding */
int numRows;                        /* number of block rows in image */
int numCols;                        /* number of block columns in image */
int row, col;                       /* location of a block */
int i0;                             /* index of first pixel in block */
double cor;                         /* correlation between block and wr */
int bitNum;                         /* number of bit embedded in block */
int zm;                             /* index in sublattice for message */
double wa;                          /* one element of added pattern */
int x, y;                           /* position within block */

/* Normalize the description of the lattice. */
/* Normalisation du pattern wr0:
        wr = wr0/|wr0|
   et beta = beta0.|wr0|                    */
beta = NormalizeLatticeDesc( beta0, wr0, wr );

/* Find number of blocks in image. */
numRows = height / 8;
numCols = width / 8;

/* Encode message. */
PadAndCodeMsg( m, mLen, mc, numRows * numCols ); // Codage convolutif (on obtient mc)

...
```

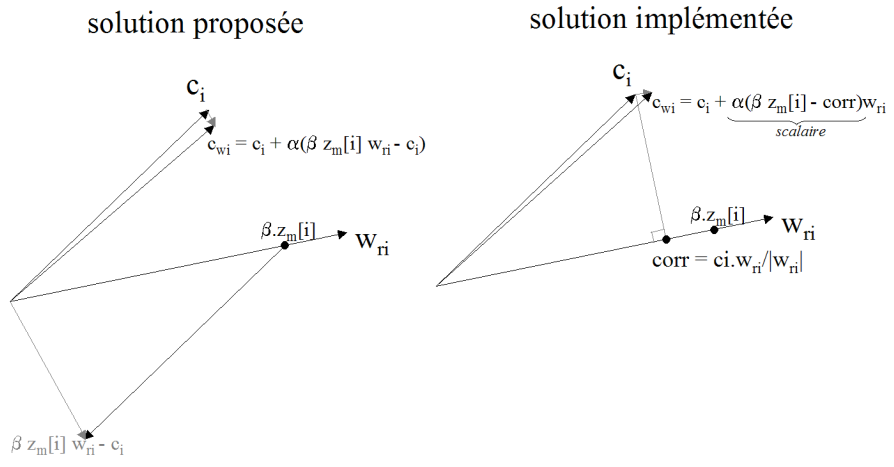

Exemple de schéma complet basé sur QIM

```
/* Embed coded bits. */
bitNum = 0;
for( row = 0; row < numRows; row = row + 1 )
  for( col = 0; col < numCols; col = col + 1 )
  {
    /* Find the first pixel of this block. */
    i0 = row * 8 * width + col * 8;

    /* Correlate this block with the reference pattern. */
    cor = 0;
    for( y = 0; y < 8; y = y + 1 )
      for( x = 0; x < 8; x = x + 1 )
        cor = cor + c[ i0 + y * width + x ] * wr[ y * 8 + x ];

    /* Find the index of the closest point in the sublattice for this message. */
    // EQUIVALENT A LA FORMULE VU PRECEDEMENT CAR |wr| = 1
    zm = (int)floor( (cor - mc[ bitNum ] * beta) / (2 * beta) + .5 ) * 2 + mc[ bitNum ];
...
```


Exemple de schéma complet basé sur QIM

```
    /* Modify the block so that its correlation with the reference
       pattern, scaled by alpha, will be close to zm. */
    for( y = 0; y < 8; y = y + 1 )
      for( x = 0; x < 8; x = x + 1 )
      {
        /* Find one element of the added pattern. */
        wa = alpha * (beta * zm - cor) * wr[ y * 8 + x ];
        // pas exactement équivalent à la formule (meilleur resultat)
        // (on effectue la pondération alpha dans le domaine corrélation;
        //  puis on rétroprojecte dans le domaine marque)

        /* Add wa to the image with clipping and rounding. */
        c[ i0 + y * width + x ] = WMTools::ClipRound( c[ i0 + y * width + x ] + wa );
      }

    /* Go on to the next bit. */
    bitNum = bitNum + 1;
  }

  delete[] mc;
}
```


Illusration de l'insertion



Exemple de schéma complet basé sur QIM - Détecteur

```
/*-------------------------------------------------------------------------*
 | D_LATTICE -- detect a message embedded with lattice-coded watermarking  |
 | Arguments:                                                              |
 |   c -- image                                                            |
 |   width -- width of image                                               |
 |   height -- height of image                                             |
 |   beta0 -- lattice spacing (as a multiple of the length of wr0)         |
 |   wr0 -- basic reference mark (8x8 pattern)                             |
 |   m -- where to store detected message                                  |
 | Return value:                                                           |
 |   length of detected message (number of bits)                           |
 *-------------------------------------------------------------------------*/
int WM_LATTICE::D_LATTICE( unsigned char *c, int width, int height,
                           double beta0, double *wr0, int *m ) {
  double beta;                      /* value of beta0 after normalizing */
  double wr[ 64 ];                  /* value of wr0 after normalizing */
  int mc[ width * height ];         /* coded message */
  int mcLen;                        /* length of coded message */
  int mLen;                         /* length of decoded message */
  int numRows;                      /* number of block rows in image */
  int numCols;                      /* number of block columns in image */
  int row, col;                     /* location of a block */
  int i0;                           /* index of first pixel in block */
  double cor;                       /* correlation between block and wr */
  int bitNum;                       /* number of bit embedded in block */
  int z;                            /* index in lattice */
  int x, y;                         /* position within block */
```


Exemple de schéma complet basé sur QIM - Détecteur

```
...

/* Normalize the description of the lattice. */
/* Normalisation du pattern wr0:
       wr = wr0/|wr0|
   et beta = beta0.|wr0|                    */
beta = NormalizeLatticeDesc( beta0, wr0, wr );

/* Find number of blocks in image. */
numRows = height / 8;
numCols = width / 8;

...
```


Exemple de schéma complet basé sur QIM - Détecteur

```
  /* Detect coded bits. */
  bitNum = 0;
  for( row = 0; row < numRows; row = row + 1 )
    for( col = 0; col < numCols; col = col + 1 )
    {
      /* Find the first pixel of this block. */
      i0 = row * 8 * width + col * 8;

      /* Correlate this block with the reference pattern. */
      cor = 0;
      for( y = 0; y < 8; y = y + 1 )
        for( x = 0; x < 8; x = x + 1 )
          cor = cor + c[ i0 + y * width + x ] * wr[ y * 8 + x ];

      /* Find the index of the closest point in the lattice */
      z = (int)floor( cor / beta + .5 );

      /* The least significant bit of z is the watermark bit in this block */
      mc[ bitNum ] = z & 1;

      /* Go on to the next bit. */
      bitNum = bitNum + 1;
    }
  mcLen = bitNum;
  /* Decode the detected bit sequence. */
  mLen = WM_TRELLIS::TrellisDecode( mc, mcLen, m );
  return mLen;
}
```

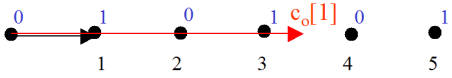

Avantages et problèmes

- simple mais :
- Les lattice ortohongonales n'assurent pas une distance égale entre mots de code.
- Les lattice hexagonales assurent une distance égale entre mots de code mais regroupent les mots dans la même région ; D'autres lattices existent...
- Faible contre les attaques valumétriques. On peut contre-carrer ceci en insérant dans un domaine non sensible aux attaques valumétriques (exemple de domaine d'insertion : corrélation normalisée)

Remarque : Least-significant-bit Watermarking

Le principe de l'insertion LSB est de substituer le bit non significatif du pixel i par le bit du message à insérer. Cette technique peut être vue comme un tatouage informé basé lattice. Cette technique est bien sûr non robuste mais peut être utilisée en stéganographie ou bien pour le tatouage fragile.

Pour le 1er pixel : $c_o[1]$



Plan

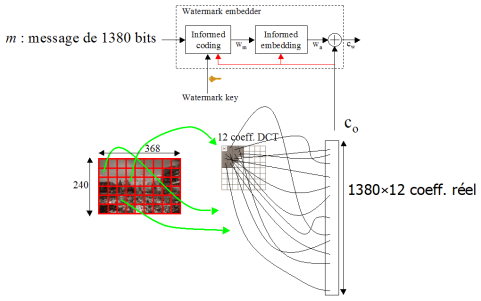
Codage dirty-paper trellis code

"**Applying Informed Coding and Informed Embedding to Design a Robust, High Capacity Watermark**". M. L. Miller, G. J. Doërr and I. J. Cox, In IEEE Transactions on Image Processing, 13(6) :792-807, 2004.

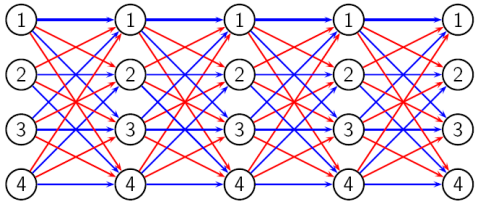
On souhaite embarquer un message m de taille L (ex : $L = 1380$ bits) dans une image de taille N (ex : $N = 240 \times 368$).

Codage dirty-paper trellis code

On souhaite embarquer un message m de taille L (ex : $L = 1380$ bits) dans une image de taille N (ex : $N = 240 \times 368$).
Soit c_o le vecteur hôte (espace d'insertion) contenant les 12 premiers coefficients ACs des blocs DCT 8×8 mélangés ; le vecteur c_o est donc composé de $12 \times N/64$ coefficients réels (On a $12 \times N/64 = 12 \times L = 12 \times 1380$)



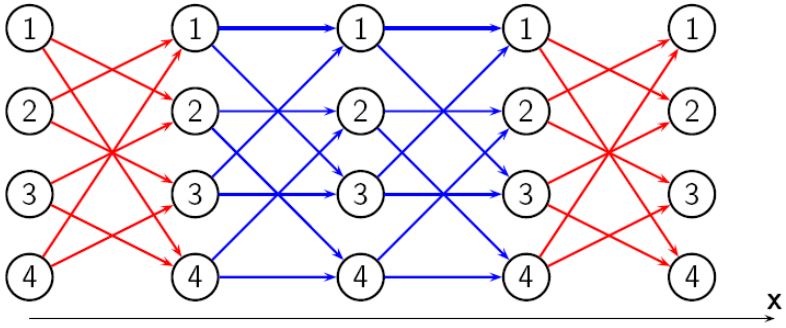
Définition du treillis utilisé pour le codage informé (étape1)



Définition du treillis :

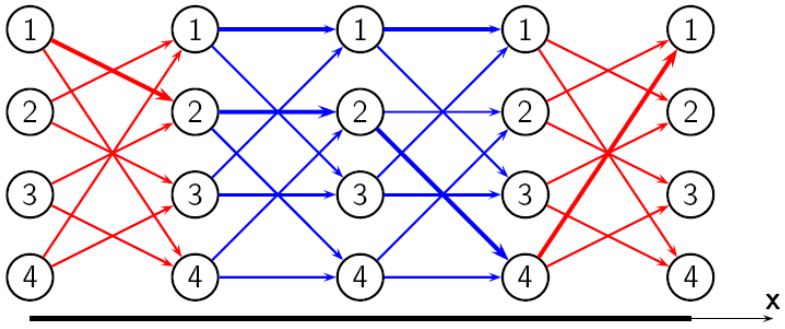
- 64 états,
- 64 arcs sortant d'un état,
- L étapes (exemple $L = 1380$),
- arc bleu = entrée 0, arc rouge = entrée 1,
- chaque arc est valué (sortie du codage) par une séquence pseudo-aléatoire de 12 coefficients réels.

Illustration du codage informé (étape 1) - élagage du treillis



Encoding of $\mathbf{m} = (1001)$

Illustration du codage informée (étape 1) - codage

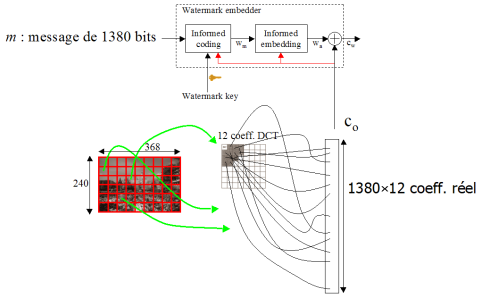


Encoding of $\mathbf{m} = (1001)$

On détermine le chemin le plus corrélé (produit scalaire) entre les arcs de sortie et le vecteur c_o (noté x ici) : Algorithme de Viterbi modifié.

Les étapes du schéma de Miller, Doërr et Cox

1. Affectuer le **codage informé** en traversant un treillis élagué. Le chemin de corrélation (produit scalaire) maximum entre c_o et les valeurs des arcs de sortie du treillis permet d'obtenir le code (concaténation des valeurs de sortie des arcs). Ce chemin est obtenu en reprenant le principe de décodage Viterbi.



Les étapes du schéma de Miller, Doërr et Cox

1. Affectuer le **codage informé** en traversant un treillis élagué. Le chemin de corrélation (produit scalaire) maximum entre c_o et les valeurs des arcs de sortie du treillis permet d'obtenir le code (concaténation des valeurs de sortie des arcs). Ce chemin est obtenu en reprenant le principe de décodage Viterbi.
2. Effectuer **l'insertion informée** : Dans l'article, l'insertion s'effectue de manière itérative (approche Monte Carlo) en 1-"attaquant" le signal marqué et en 2-"déplacant" le signal marqué si l'attaque a réussi. Un masque psychovisuel (Watson) est également utilisé. Le problème d'une approche de type Monte Carlo est sa complexité calculatoire.

Travaux autour de l'approche...

- Dans le papier [Lin et al. 2005], une approche moins complexe calculatoirement a été proposée pour rendre plus attrayant le schéma complet. *[Lin et al. 2005] "An efficient algorithm for informed embedding of dirty-paper trellis codes for watermarking", L. Lin, I. J. Cox and G. Doerr, IEEE Int. Conf. on Image Processing, 2005.*
- D'autres travaux ont également été proposés pour comprendre les mécanismes de robustesse et de définition du treillis,
- Le schéma est celui qui a été utilisé pour le compétition BOWS-1 (achevée en juin 2006)

Plan

Code basé syndrome

Ici, les syndromes (cf. codes correcteurs) sont utilisés d'une façon étonnante : le syndrome porte le message.
Problème : pas très robuste (avec cette utilisation là) et assez marginal. Probablement plus interessant pour la steganographie (Wet Paper Code).

"On the duality between distributed source coding and data hiding," J. Chou, S. S. Pradhan, and K. Ramchandran, in Proc. Thirty-third Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, USA, Oct. 1999, vol. 2, pp. 1503-1507.

Schéma à étalement de spectre informé : BPSK spread-spectrum watermarking modulations

(BPSK : Binary Phase Shift Keying)
Improved Spread Spectrum ; Natural Watermarking, Circular Watermarking, CHI2 Watermarking.

"**Improved spread spectrum : a new modulation technique for robust watermarking**.", H. S. Malvar and D. F. Rencio. IEEE Transaction on Signal Processing, 53 :898-905, Apr. 2003.

"**Natural Watermarking : a secure spread spectrum technique for WOA**" Patrick Bas, and François Cayre, Information Hiding 2006, pp.1-14, 4437.

"**Techniques sûres de tatouage pour l'image**", B. Mathon, P. Bas, F. Cayre, CORESA'2007, COmpression et REprésentation des Signaux Audiovisuels, 8-9 Novembre 2007, Montpellier, France.

"**Practical performance analysis of secure modulations for WOA spread-spectrum based image watermarking**", B. Mathon, P. Bas, F. Cayre. ACM'2007, Multimedia and Security Workshop, 20-21 September 2007, Dallas, Texas, USA.

"**Distortion Optimization of Model-Based Secure Embedding Schemes for Data-Hiding**" B. Mathon, P. Bas, F.

Sujets de stage 2008-2009... pub !

- "Tatouage robuste aux attaques de désynchronisations" (résister au "print-and-scan" ou rognage et être sûr),
- "Attaque de systèmes de tatouage" (attaque des DPTC, ou attaque de BA de BOWS-2, ou analyse des techniques récentes en vue de la proposition de schémas sûrs).

Article à lire pour l'examen 2008-2009

- "**A Regression-Based Restoration Technique for Automated Watermark Removal**", Andreas Westfeld, TU Dresden, Multimedia & Security ACM Workshop MMSEC2008, Oxford, United Kingdom, 22-23 September 2008.
- "**Self-Synchronizing Robust Texel Watermarking in Gaussian Scale-Space**", Mathias Schlauweg, Dima Pröfrock, Benedikt Zeibich, Erika Müller, Multimedia & Security ACM Workshop MMSEC2008, Oxford, United Kingdom, 22-23 September 2008.