THEORIE DE L'INFORMATION

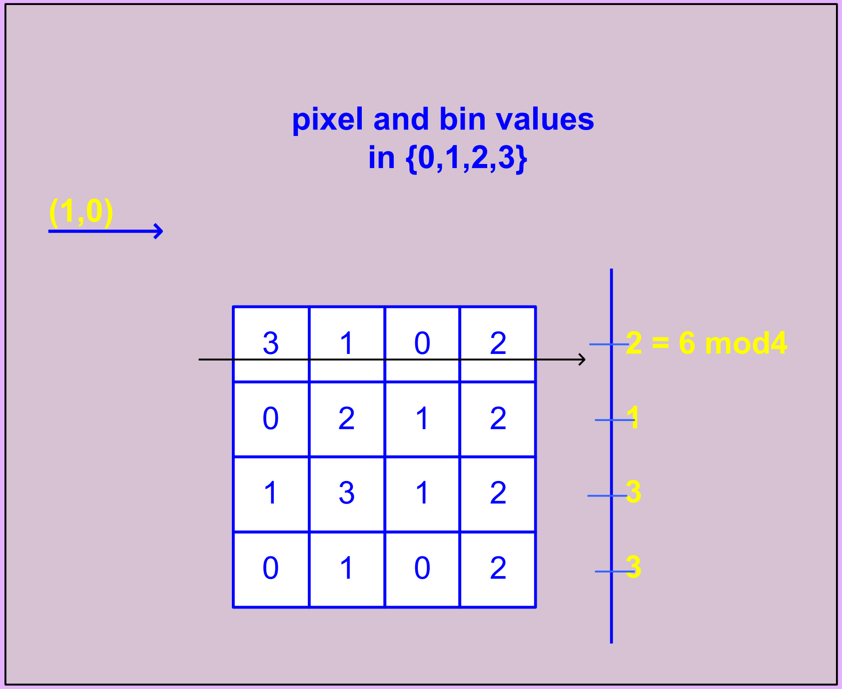
chapitre 5

Transformée Mojette

INFO 3

Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$



Transformée LINEAIRE Mojette

$$proj_f(b,p,q) = \sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl) \right)$$

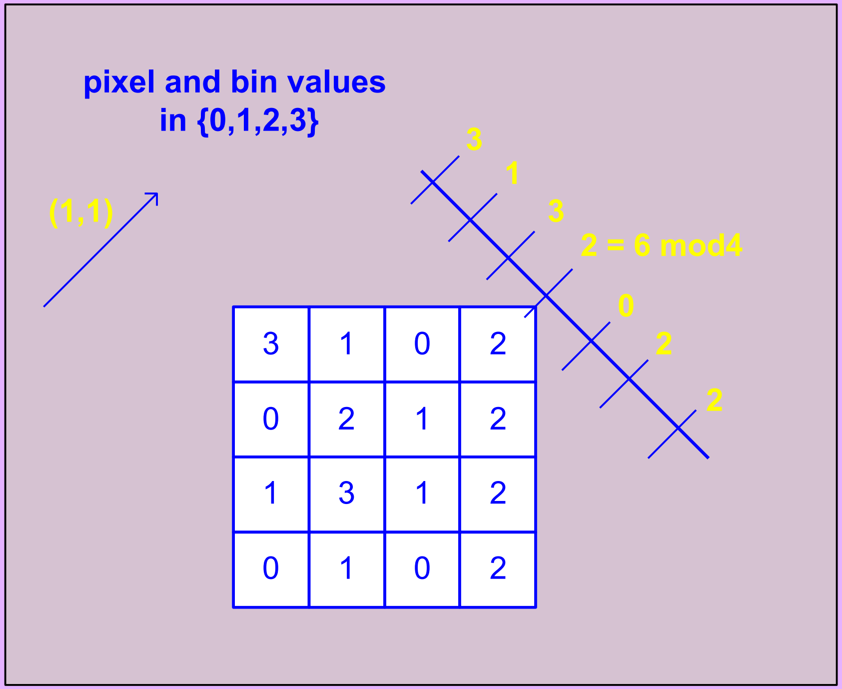
$$\Delta(b+qk-pl) = 0 \; si \; b+qk-pl \neq 0$$

$$\Delta(b+qk-pl) = 1 \; si \; b+qk-pl = 0$$

Def droite

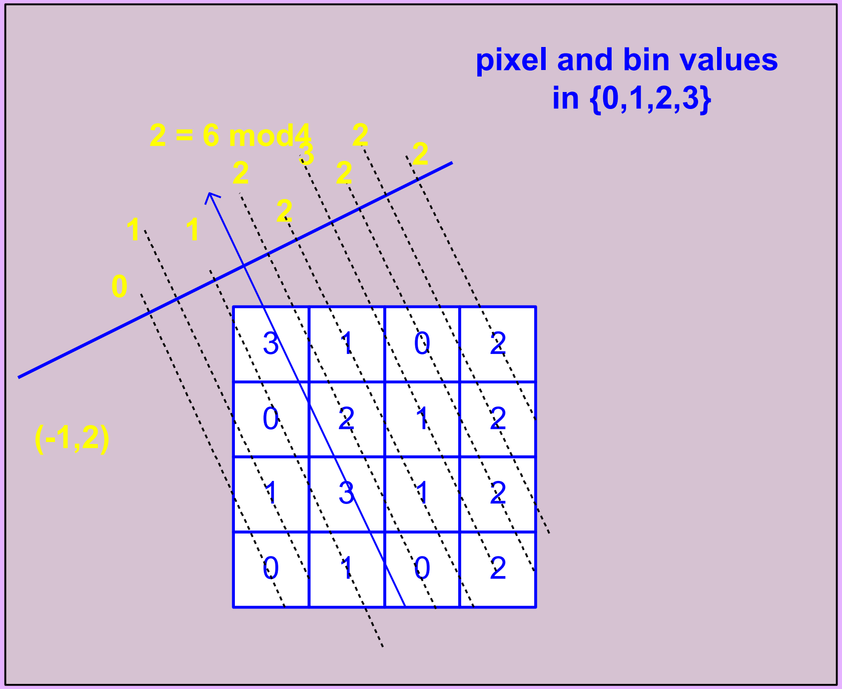
Transformée Mojette

$$proj_f(b,p,q)=\sum_k\sum_l\left(f(k,l)\Delta(b+qk-pl)\right)$$



Transformée Mojette

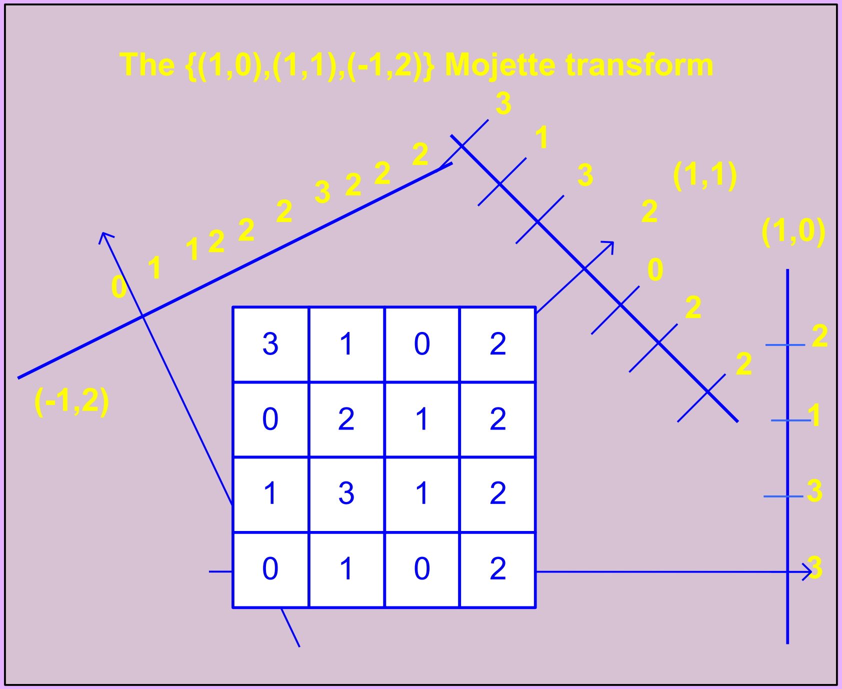
$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$

1 image, 16 pixels → 3 projections, 21 bins

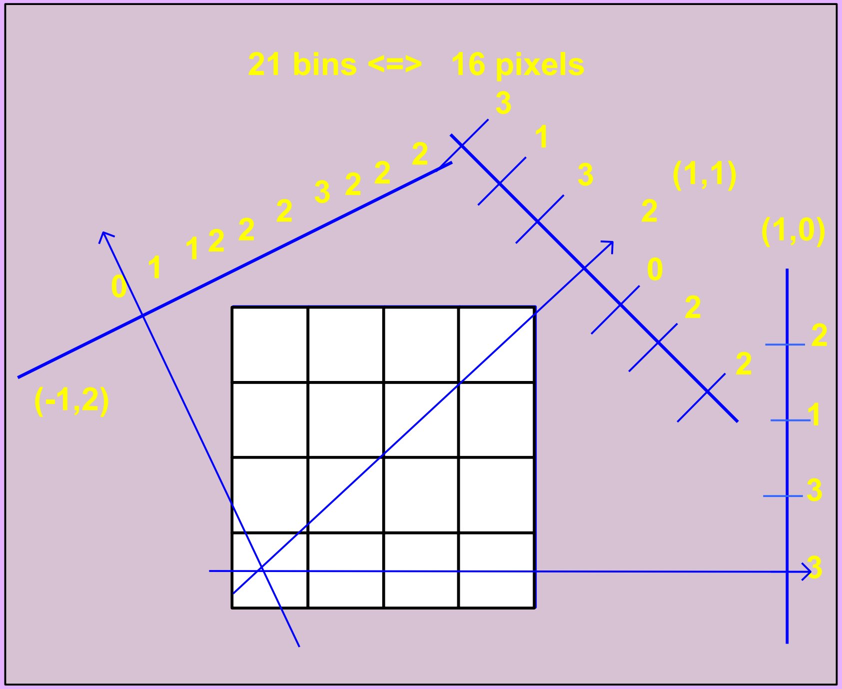


$$\mathrm{Red}=\frac{\#\mathrm{bins}}{\#\mathrm{pixels}}-1$$

Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

1 image, 16 pixels ⇐ 3 projections, 21 bins

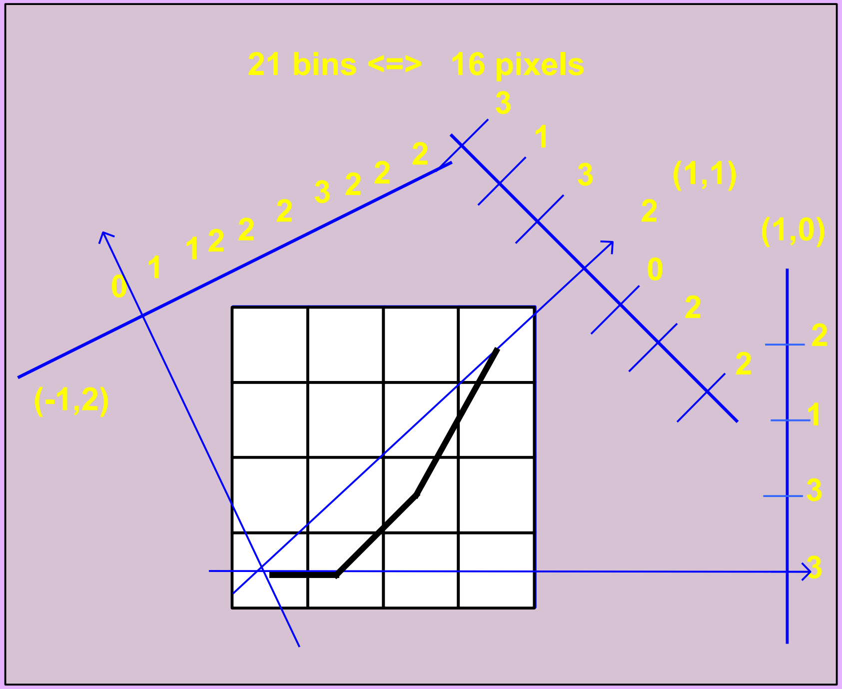


$$\mathrm{Red}=\frac{\#\mathrm{bins}}{\#\mathrm{pixels}}-1$$

Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

Possible reconstruction ?

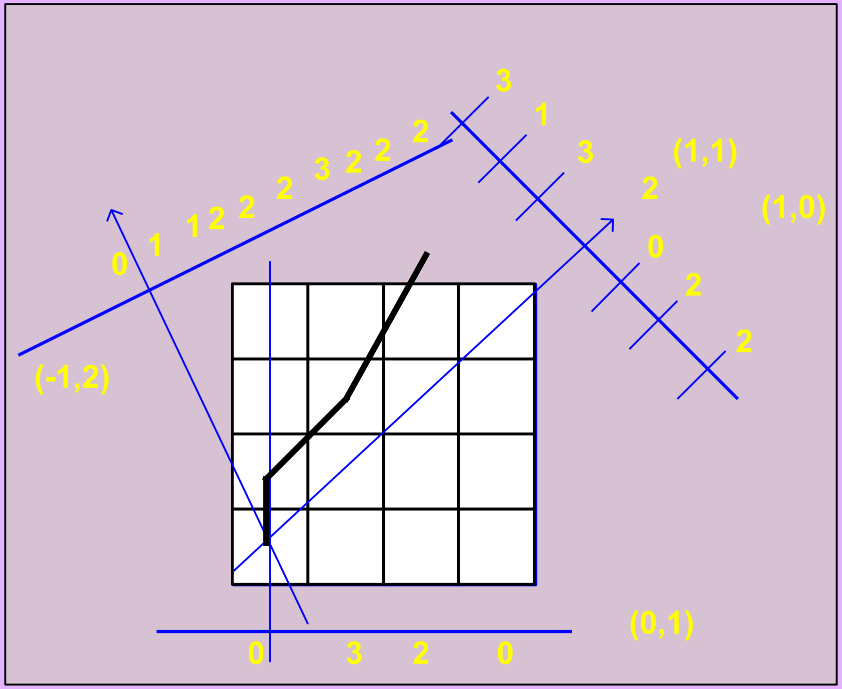


Non!

Transformée Mojette

$$proj_f(b,p,q) = \sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl) \right)$$

Reconstruction ?

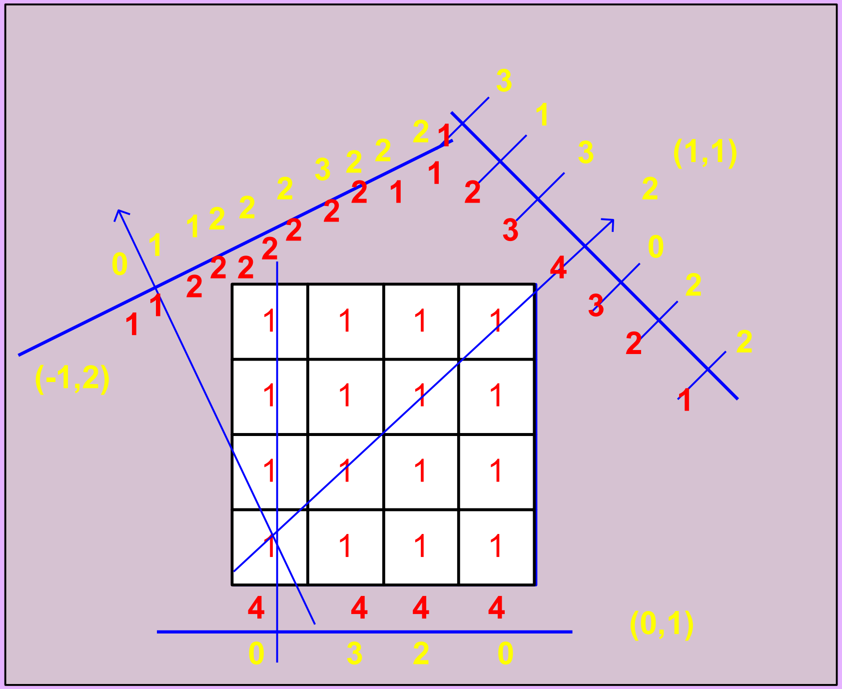


Oui !

Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$

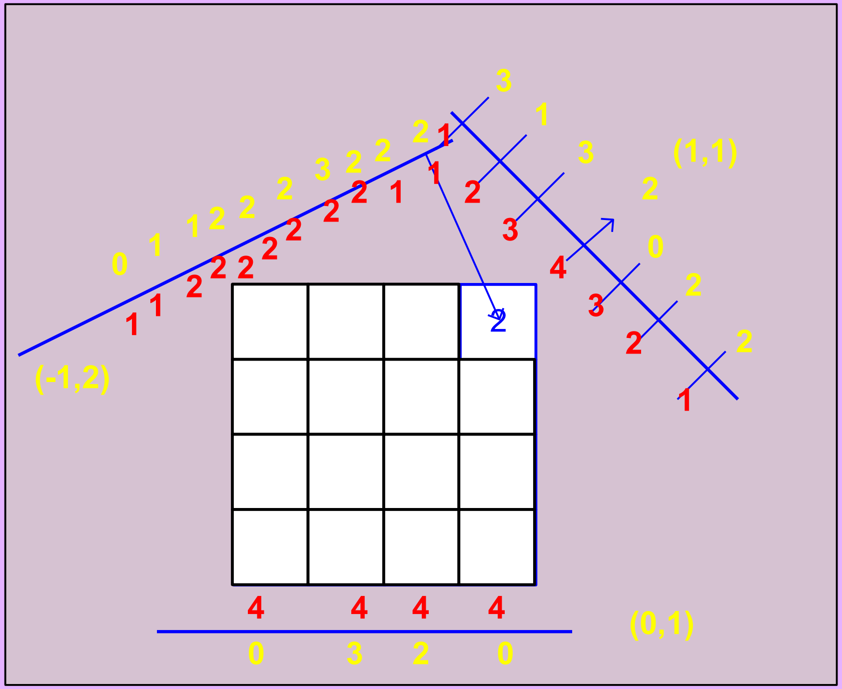
En rouge = T Mojette de la fonction support



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

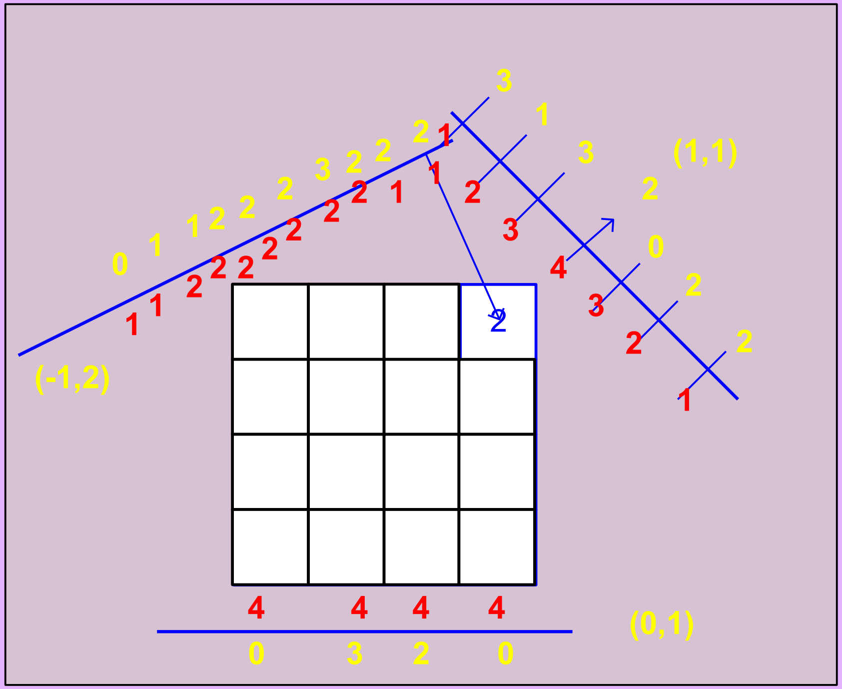
N'importe quelle correspondance 1 pixel to 1 bin peut être utilisée



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$

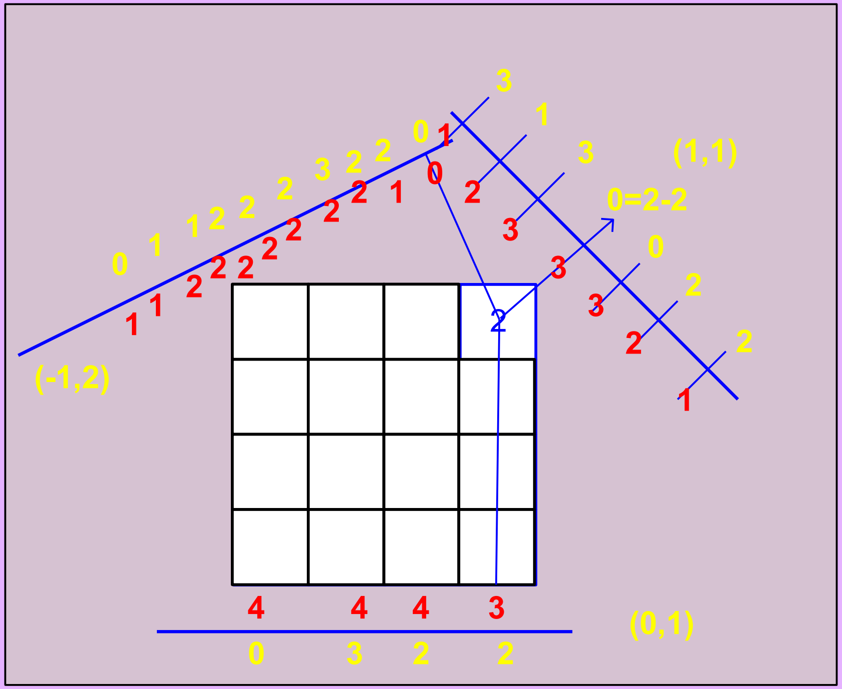
Pixel 1, Step 1 : rétroprojection



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

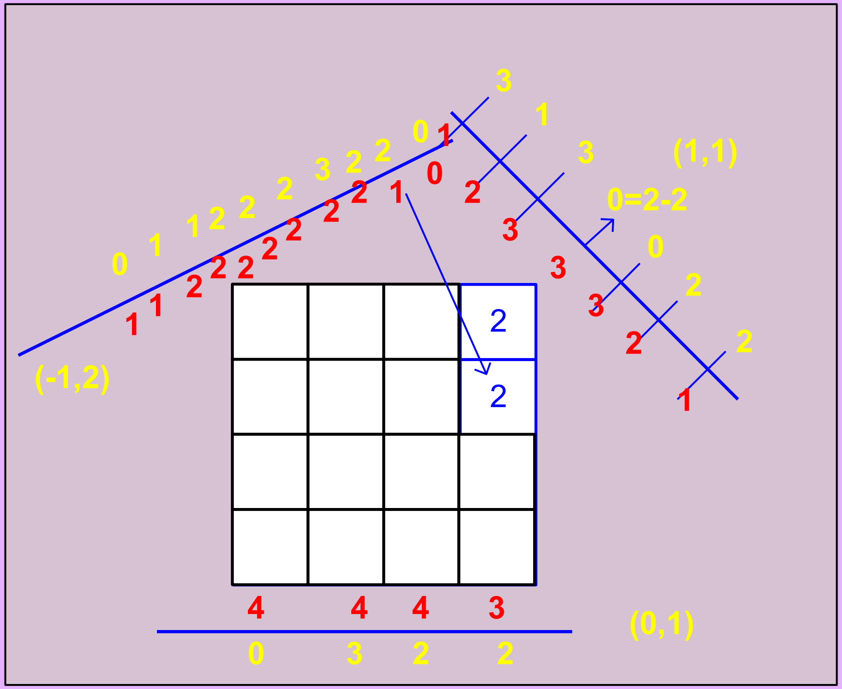
Pixel 1, Step 2 : mise à jour des projections



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$

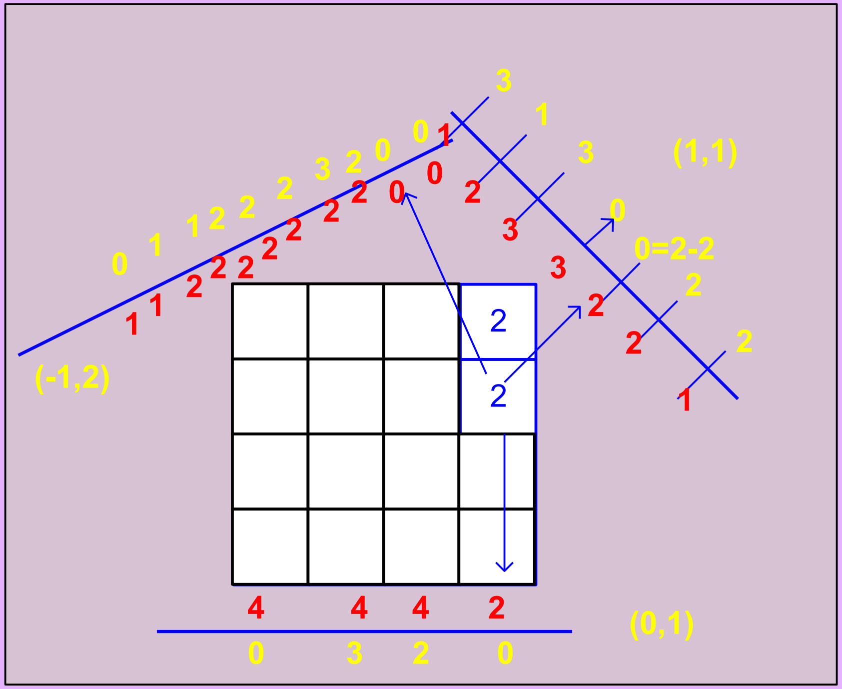
Pixel 2, Step 1



Transformée Mojette

$$proj_f(b,p,q)=\sum_k\sum_l\left(f(k,l)\,\Delta(b+qk-pl)\right)$$

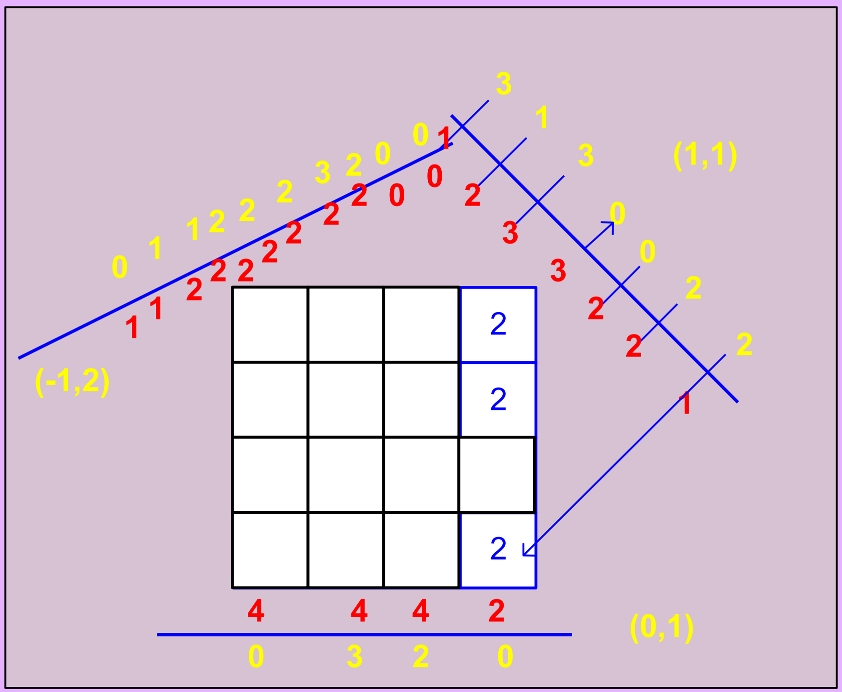
Pixel 2, Step 2



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

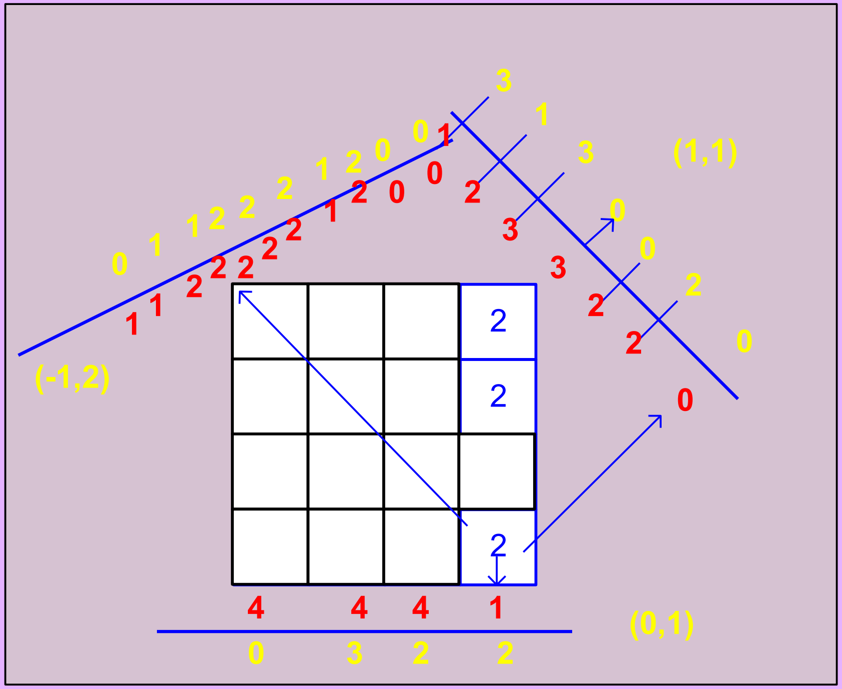
Pixel 3, Step 1



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$

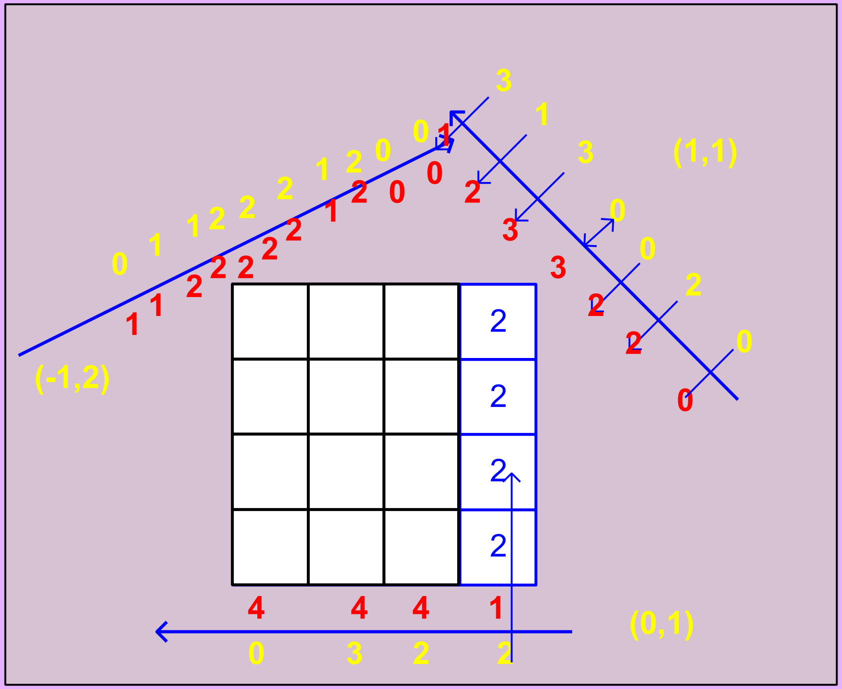
Pixel 3, Step 2



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

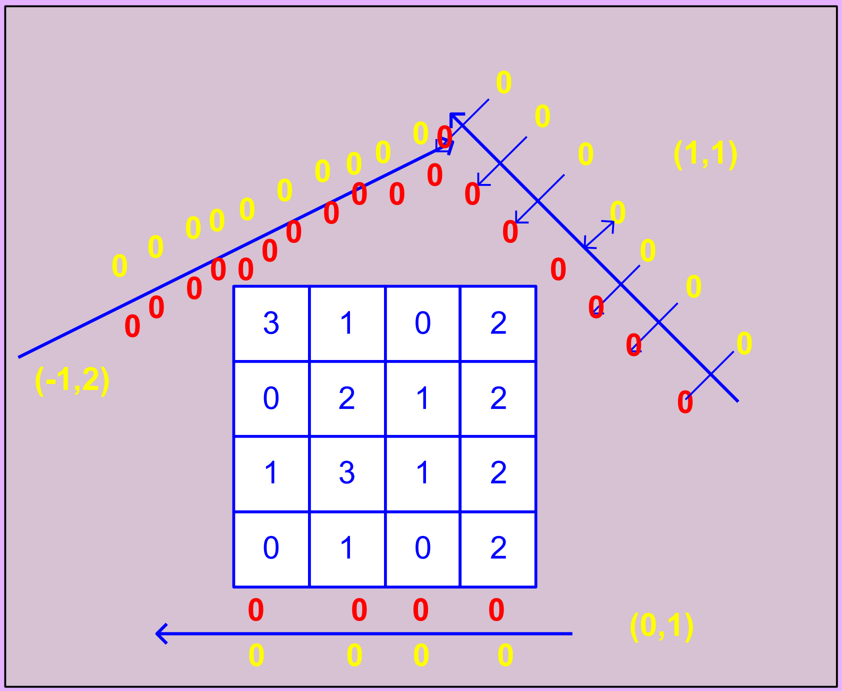
Pixel 4



Transformée Mojette

$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\,\Delta(b+qk-pl)\right)$$

Pixel 16



Algorithme Mojette inverse

Algorithm M^{-1} (Ph.D. N.Normand 97)

```
begin
        pour chaque pixel de la forme
                (0) trouvez 1 relation pixel-bin
                (1) rétroprojete valeur du  bin dans pixel
                (2) pour chaque angle
                        mettre à jour le bin
                    fin pour
         fin pour
end
```

Transformée Mojette

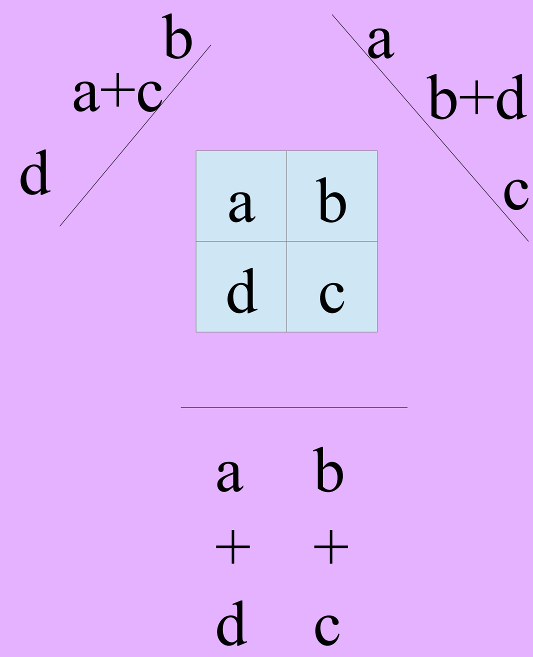
$$proj_f(b,p,q)=\sum_k \sum_l \left(f(k,l)\Delta(b+qk-pl)\right)$$

Utilisation en code correcteur :
si besoin de N projections pour reconstruire, on en ajoute P de sorte que n'importe quel ensemble de n dans N+P reconstruise

Code correcteur Mojette

Exemple d'utilisation :

2 des 3 projections reconstruisent



Somme des bins d'1projection

=

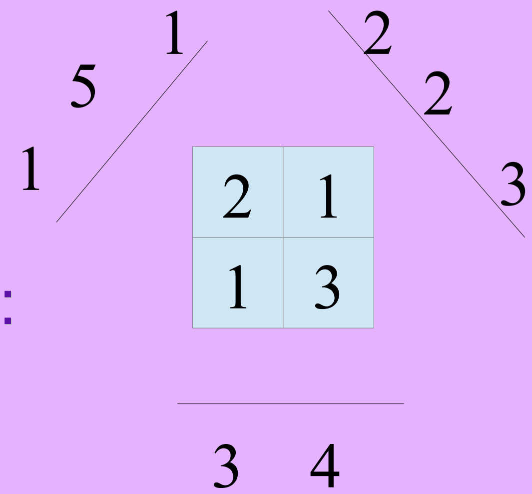
somme des pixels

Code correcteur Mojette

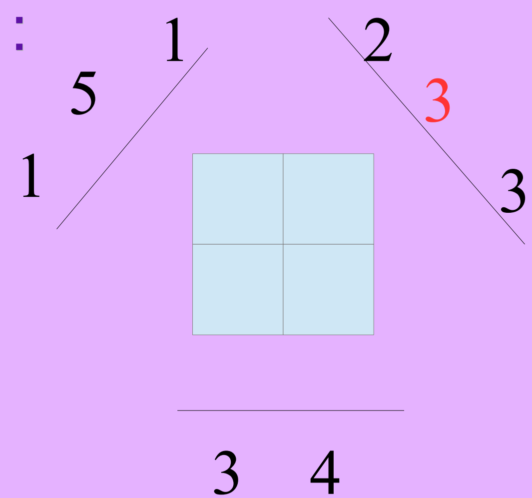
Exemple d'utilisation :

2 des 3 projections reconstruisent

J'envoie :



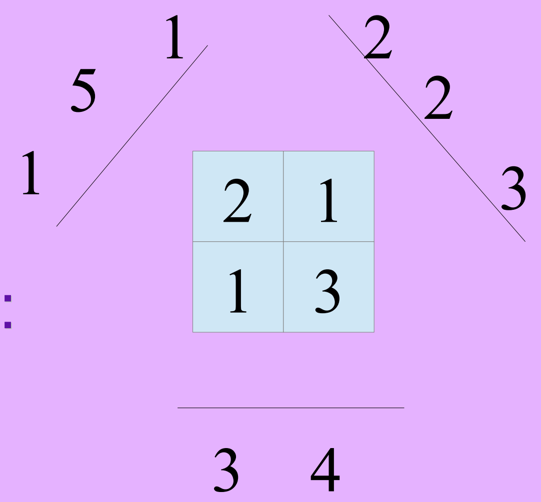
Je recois :



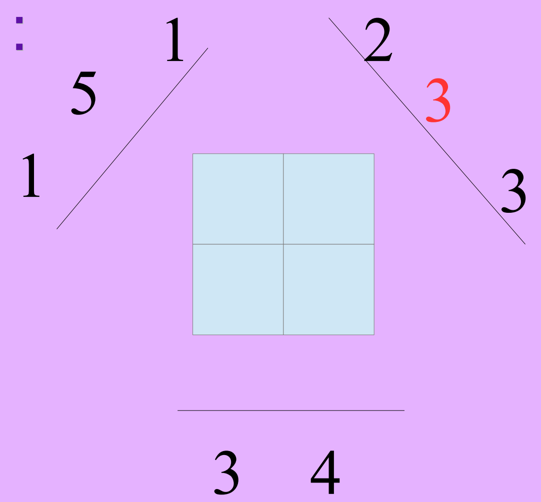
Code correcteur Mojette

Somme sur projections :
7 , 7 et 8
=> indique proj fausse

J'envoie :

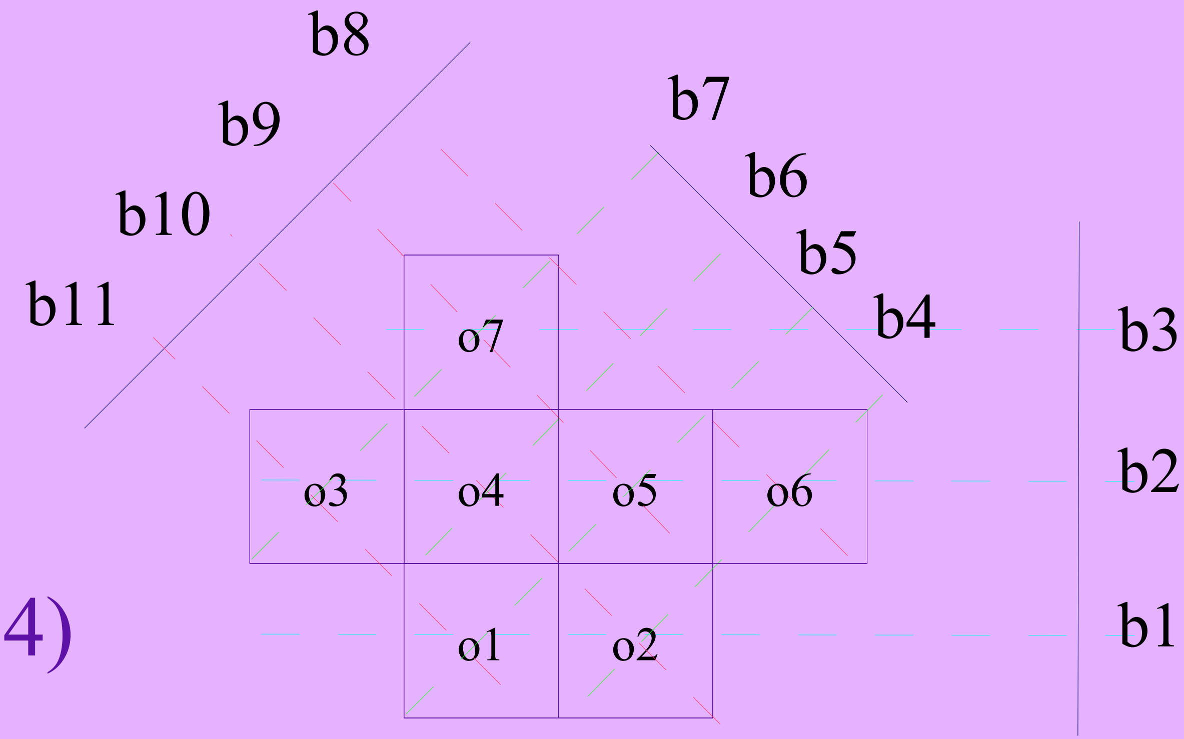


Je recois :



Code correcteur Mojette

Moj(11,7)



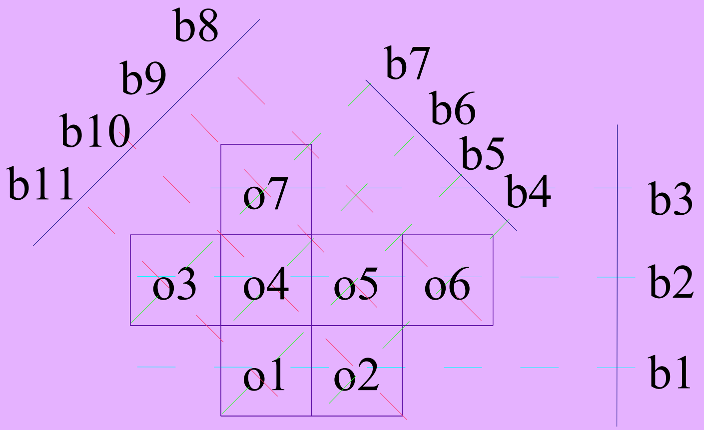
entre

Hamming H(7,4)

et H(15,11)

Envoi : les 3 proj
En parallèle

$$\left(\Sigma_0 = \sum_{i=1}^{7} o_i\right)$$

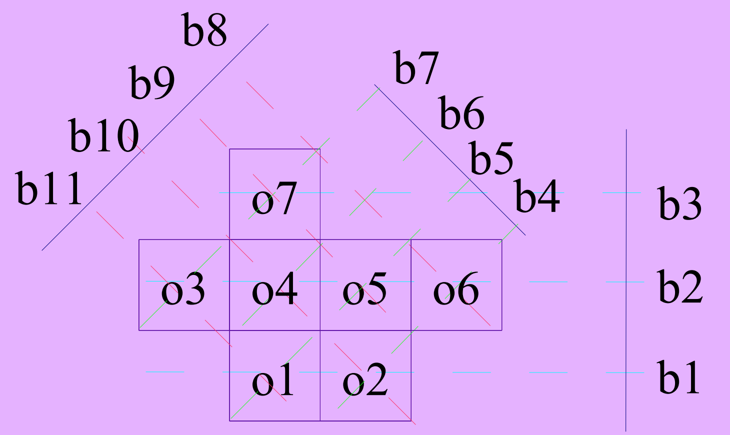


$$\Sigma_1 = b_1 + b_2 + b_3$$
$$\left(\Sigma_2 = b_4 + b_5 + b_6 + b_7\right)$$
$$\Sigma_3 = b_8 + b_9 + b_{10} + b_{11}$$

$$\left(\Sigma_0 = \Sigma_1 = \Sigma_2 = \Sigma_3\right)$$

Réception :

On calcule



$$\Sigma_1 = b_1 + b_2 + b_3$$

$$\left(\Sigma_2 = b_4 + b_5 + b_6 + b_7 \right)$$

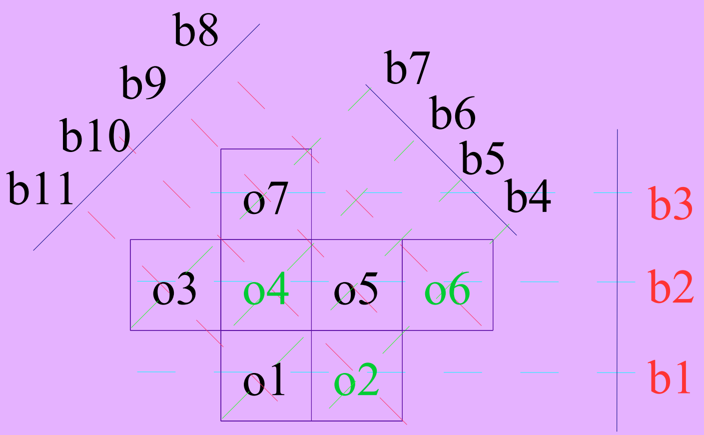
$$\Sigma_3 = b_8 + b_9 + b_{10} + b_{11}$$

Cas 0 $\left(\Sigma_1 = \Sigma_2 = \Sigma_3 \right)$ => pas d'erreur

Réception :

Cas 1 $\Sigma_1 \neq (\Sigma_2 = \Sigma_3)$

$$\Sigma_1 = b_1 + b_2 + b_3$$
$$\left(\Sigma_2 = b_4 + b_5 + b_6 + b_7 \right)$$
$$\Sigma_3 = b_8 + b_9 + b_{10} + b_{11}$$

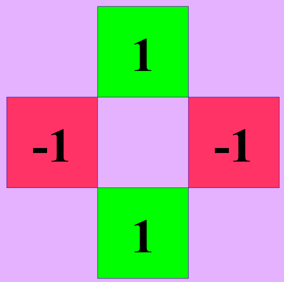


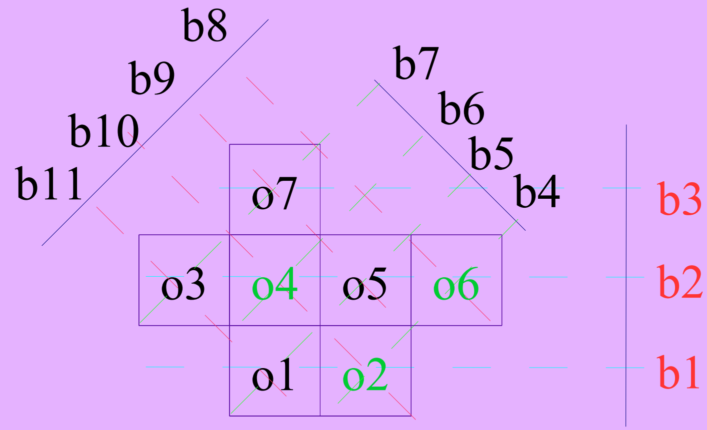
=> erreur sur proj (1 0) , on utilise d'abord les 2 autres proj
=> reconstruit o2 o4 o6

Réception :

Cas 1 $\Sigma_1 \neq (\Sigma_2 = \Sigma_3)$

$$o_1 + o_3 = b_{11}$$
$$\left(o_1 + o_5 = b_5 \right)$$
$$o_5 + o_7 = b_9$$
$$o_3 + o_7 = b_7$$



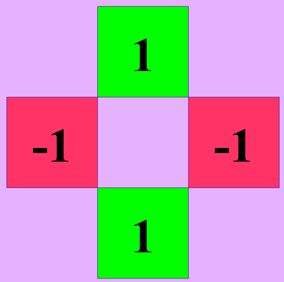


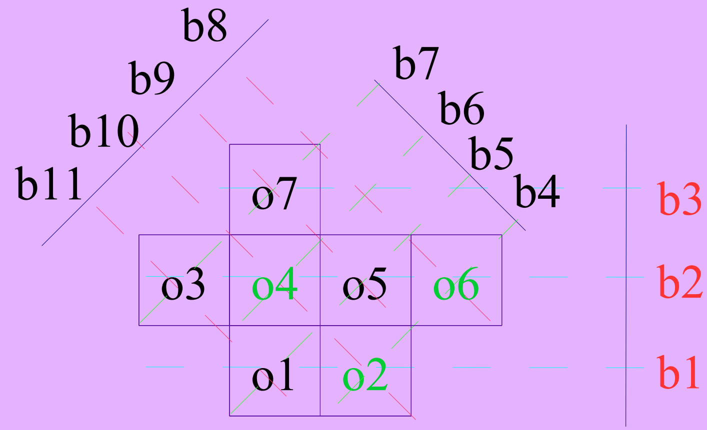
=> on sait que 2 valeurs sur 3 sont bonnes dans (b1 b2 b3)

Réception :

Cas 1 $\Sigma_1 \neq (\Sigma_2 = \Sigma_3)$

$$o_1 + o_3 = b_{11}$$
$$(o_1 + o_5 = b_5)$$
$$o_5 + o_7 = b_9$$
$$o_3 + o_7 = b_7$$





=> 1 seul de ces 3 systèmes est juste

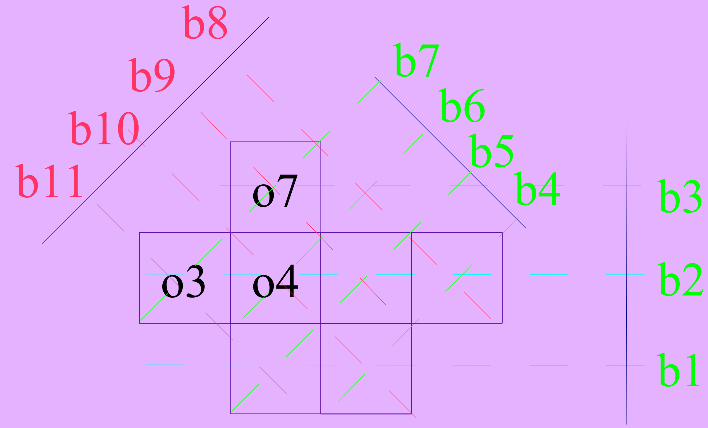
$$\bar{b}_1 = o_1 + o_2$$
$$(b_2 = o_3 + o_4 + o_5 + o_6)$$
$$b_3 = o_7$$

$$b_1 = o_1 + o_2$$
$$(\bar{b}_2 = o_3 + o_4 + o_5 + o_6)$$
$$b_3 = o_7$$

$$b_1 = o_1 + o_2$$
$$(b_2 = o_3 + o_4 + o_5 + o_6)$$
$$\bar{b}_3 = o_7$$

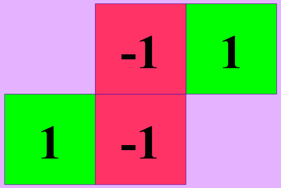
=> Localise l'erreur

Réception :



Cas 2 $\left(\Sigma_1=\Sigma_2\right)\neq\Sigma_3$

$$\begin{pmatrix} o_2+o_6=b_4 \\ o_1+o_5=b_5 \\ o_1+o_2=b_1 \\ o_3+o_4+o_5+o_6=b_2 \end{pmatrix}$$



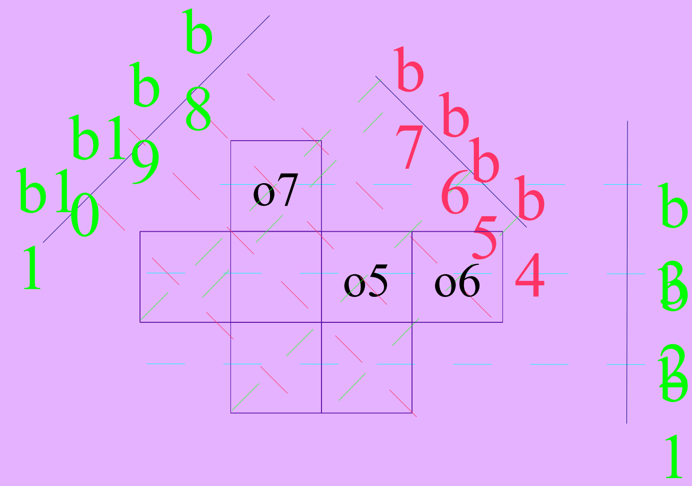
=> Localise l'erreur

=> 1 seul de ces 4 systèmes est juste

$$\begin{pmatrix} \bar{b}_8=o_6 \\ b_9=o_5+o_7 \\ b_{10}=o_2+o_4 \\ b_{11}=o_1+o_3 \end{pmatrix} \quad \begin{pmatrix} b_8=o_6 \\ \bar{b}_9=o_5+o_7 \\ b_{10}=o_2+o_4 \\ b_{11}=o_1+o_3 \end{pmatrix} \quad \begin{pmatrix} b_8=o_6 \\ b_9=o_5+o_7 \\ \bar{b}_{10}=o_2+o_4 \\ b_{11}=o_1+o_3 \end{pmatrix} \quad \begin{pmatrix} b_8=o_6 \\ b_9=o_5+o_7 \\ b_{10}=o_2+o_4 \\ \bar{b}_{11}=o_1+o_3 \end{pmatrix}$$

Réception :

Cas 3 $\left(\Sigma_1=\Sigma_3\right)\neq\Sigma_2$



$$\left(\begin{aligned} o_2+o_4&=b_{10}\\ o_1+o_3&=b_{11}\\ o_1+o_2&=b_1\\ o_3+o_4+o_5+o_6&=b_2 \end{aligned}\right)$$

-1	1	
	1	-1

=> Localise l'erreur

=> 1 seul de ces 4 systèmes est juste

$$\left(\begin{aligned} \bar{b}_4&=o_2+o_6\\ b_5&=o_1+o_5\\ b_6&=o_4\\ b_7&=o_4+o_7 \end{aligned}\right) \quad \left(\begin{aligned} b_4&=o_2+o_6\\ \bar{b}_5&=o_1+o_5\\ b_6&=o_4\\ b_7&=o_4+o_7 \end{aligned}\right) \quad \left(\begin{aligned} b_4&=o_2+o_6\\ b_5&=o_1+o_5\\ \bar{b}_6&=o_4\\ b_7&=o_4+o_7 \end{aligned}\right) \quad \left(\begin{aligned} b_4&=o_2+o_6\\ b_5&=o_1+o_5\\ b_6&=o_4\\ \bar{b}_7&=o_4+o_7 \end{aligned}\right)$$

Moj(11,7)

entre Hamming H(7,4) et H(15,11)

RedH(7,4) =(7-4)/4=3/4 =0,75

RedMoj(11,7) =(11-7)/7=4/7 =0,57...

RedH(15,11) =(15-11)/11=4/7 =0,37...

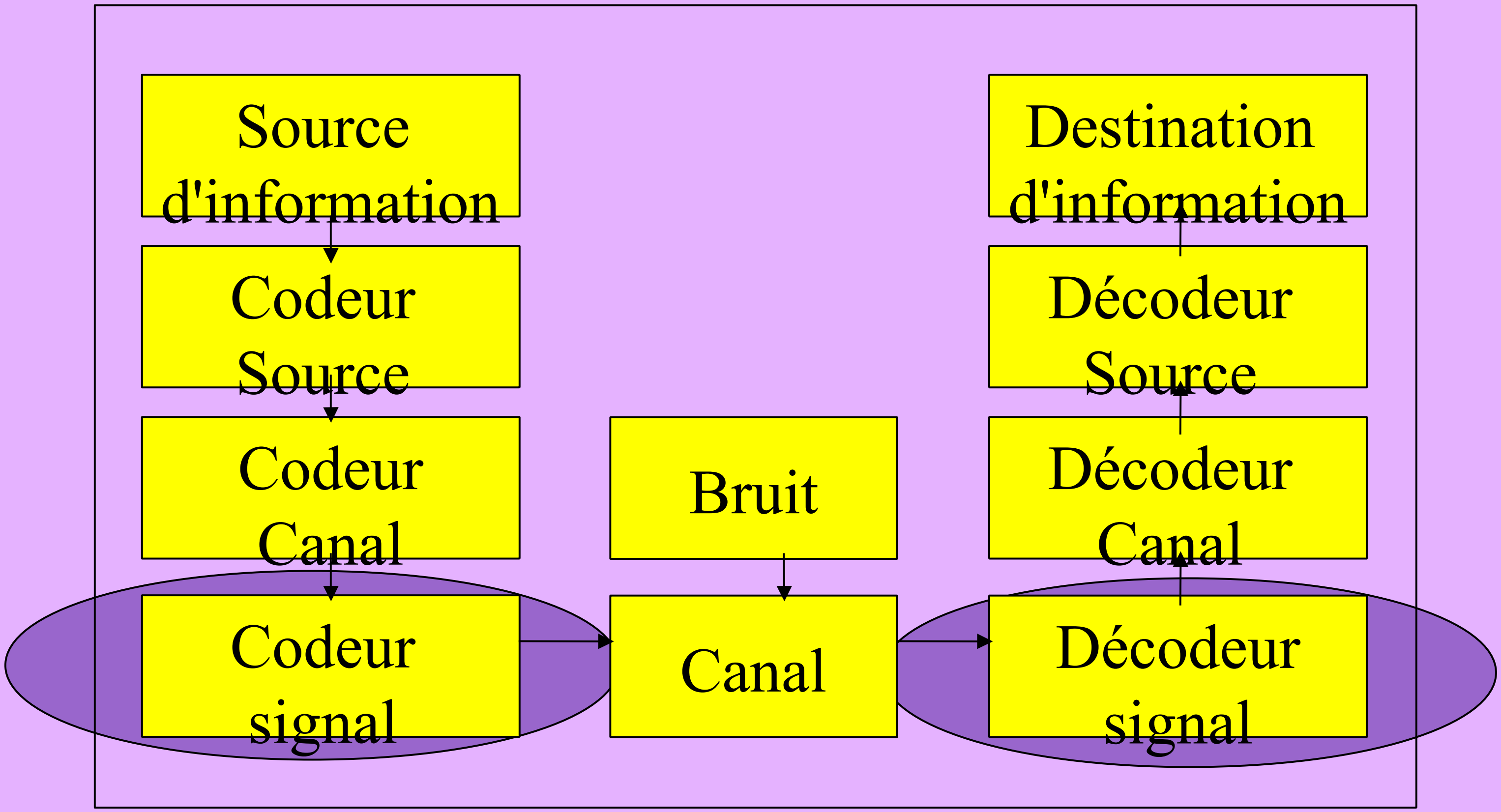
THEORIE DE L'INFORMATION

chapitre 6

Brouilleur ligne
générateur pseudo aléatoire

INFO 3

Théorie de l’ Information



1. DEF de générateur pseudo-aléatoire
à registres à décalage à rétroaction linéaire

Un registre à décalage à rétroaction linéaire, ou LFSR (acronyme de l'anglais Linear Feedback Shift Register), est un dispositif électronique ou logiciel qui produit une suite de bits qui peut être vue comme une suite récurrente linéaire sur le corps fini F2 à 2 éléments (0 et 1).
La notion a été généralisée à n'importe quel corps fini.

le corps fini F2 à 2 éléments (0 et 1)

un corps fini est un corps commutatif qui est par ailleurs fini.

+ (XOR)	0	1
0	0	1
1	1	0

X (ET)	0	1
0	0	0
1	0	1

F2 est aussi l'ensemble des valeurs de vérité classiques, 0 pour le faux, et 1 pour le vrai. L'addition est le « ou exclusif » (xor), la multiplication le « et ».

Les fonctions de F2n dans F2 sont appelées fonctions booléennes.

La disjonction (inclusive) et la négation se définissent ainsi :

$$x \, ou \, y = x + y + xy \qquad non(x) = x + 1$$

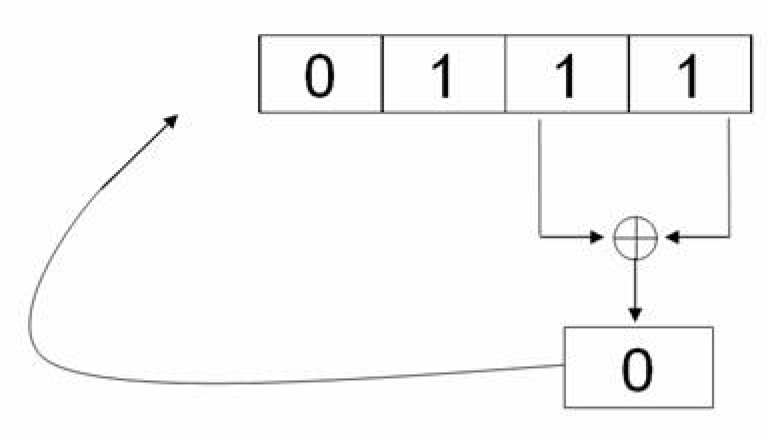
DEF de générateur pseudo-aléatoire à registres à décalage à rétroaction linéaire

La suite récurrente produite par un LFSR est nécessairement périodique à partir d'un certain rang. Les LFSR sont utilisés en cryptographie pour engendrer des suites de nombres pseudo-aléatoires. La fonction de rétroaction est alors choisie de façon à obtenir une période la plus grande possible.

DEF de générateur pseudo-aléatoire
à registres à décalage à rétroaction linéaire

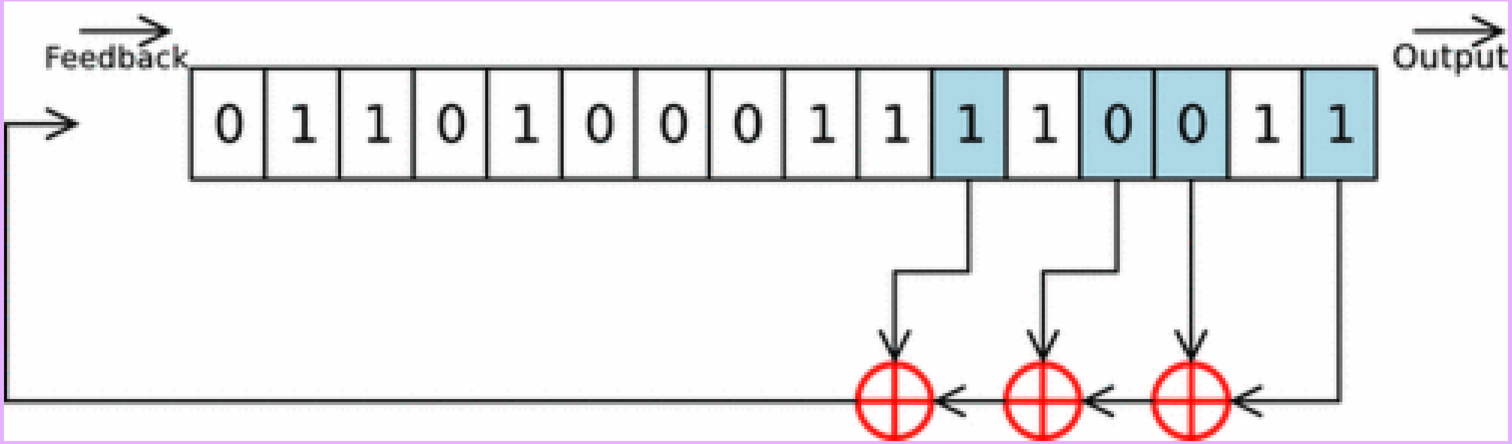
Le registre est successivement
 à la valeur 0111, 0011,
 ...

et après ??



Pour un LFSR à n bits, on peut faire en sorte d'obtenir une période maximale de 2^n–1.

DEF de générateur pseudo-aléatoire
à registres à décalage à rétroaction linéaire
https://upload.wikimedia.org/wikipedia/commons/9/99/Lfsr.gif



Et si on veut juste jouer à Pile ou Face ?

La méthode “la plus simple” est d'utiliser les polynomes primitifs dans le corps de Galois à 2 élements

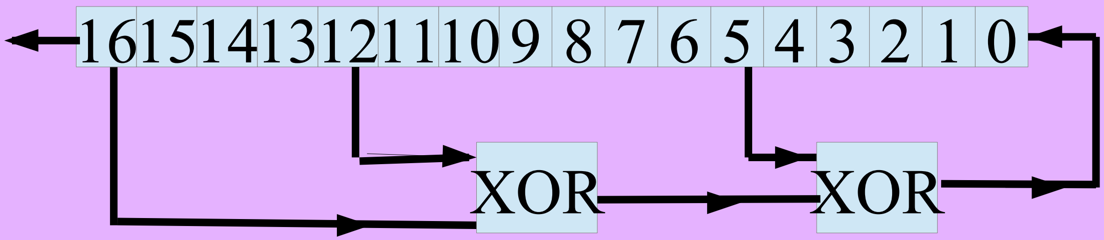
Par exemple le polynôme générateur
(CCITT CRC 16 bits) est

$$P(x) = x^{16} + x^{12} + x^{5} + x^{0}$$

Et si on veut juste jouer à Pile ou Face ?

$$P(x) = x^{16} + x^{12} + x^{5} + x^{0}$$

On fait le schéma suivant :



et on remplit les cases binaires de 0 à 15 par la racine non nulle (sinon ca ne sort que des 0 !)

la période est 2^16-1 bits

Et si on prend n'importe quel polynome ?

$$P(x) = x^2 + 1 \quad P(x) = x + 1$$

la période n'est pas de 2^16-1 bits ...

Théorie de l’ Information

- **merci**