Quantification Vectorielle

Programming Challenge

Martin D'Escrienne Yann
& Tognetti Yohann

Enoncé :

On vient de recevoir par le réseau ***une image ($I:N\times N$) de pixels en niveaux de gris ($I_{i,j}\in[0,255]$).***

Cette image a une structure de blocs ***($B_{kl}:M\times M$)*** *tel que :*

$$\begin{matrix} B_{0,0} & B_{0,1} & \dots & B_{0,K} \\ B_{1,0} & B_{1,1} & \dots & B_{1,K} \\ \dots & \dots & \dots & \dots \\ B_{K,0} & B_{K,1} & \dots & B_{K,K} \end{matrix}$$

avec ***$K=N/M$***

Ces blocs sont issus d'une liste de blocs. Ils sont indexables par ***l'entier $p\in[0,P-1]$.***

Cette liste forme ***un dictionnaire de motifs***

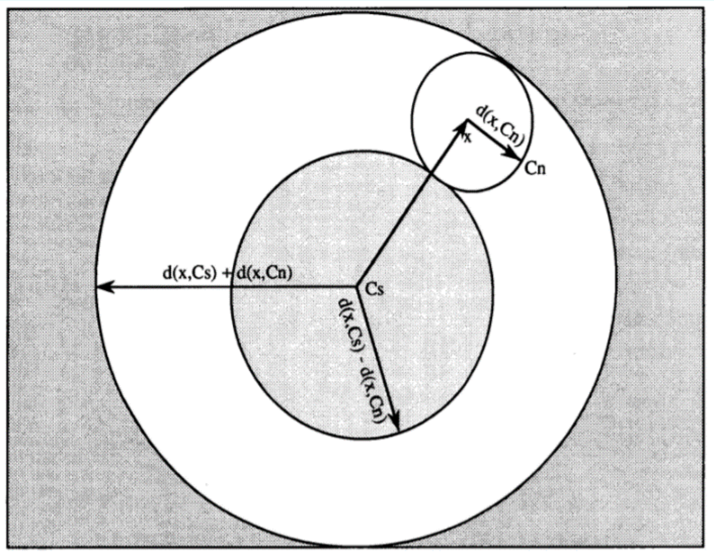
La sortie est ***la matrice des index des blocs du dictionnaire*** *utilisés dans l'image reçue.*

Il faudra en plus considérer que l'image transmise a été potentiellement ***bruitée*** *et que par conséquent, les blocs de l'image ne sont pas forcément "exactement" ceux du dictionnaire.*

Il faut donc trouver, pour chaque bloc de l'image, l'élément du dictionnaire qui est ***le plus proche au sens de la distance euclidienne****.*

Si plusieurs éléments du dictionnaire venaient à avoir la même distance à un bloc, vous choisiriez celui qui a ***l'indice le plus petit.***

Stratégie d'élimination de vecteur : Inégalité triangulaire



Lors de chaque test de vecteur du dictionnaire, notre algorithme va également potentiellement éliminés d'autres vecteur grâce au formule de l'inégalité triangulaire.

Posons :
***Cn** le vecteur actuellement pris dans le dictionnaire*
***Cs** le vecteur que nous comparons à Cn.*
***X** le vecteur de l'image.*
***Cp** le vecteur qui va potientiellement être éliminé.*

Si Cp répond à une des deux conditions suivante, alors il peut être éliminé.

$$d(c_p\ ,\ c_s) \geq d(x\ , c_s) + d(x\ , c_n) \quad (1)$$

$$d(c_p\ , c_s) \leq d(x\ , c_s) - d(x\ , c_n) \quad (2)$$

Cela peut être compris via l'image ci-dessus qui représente les deux inégalités.

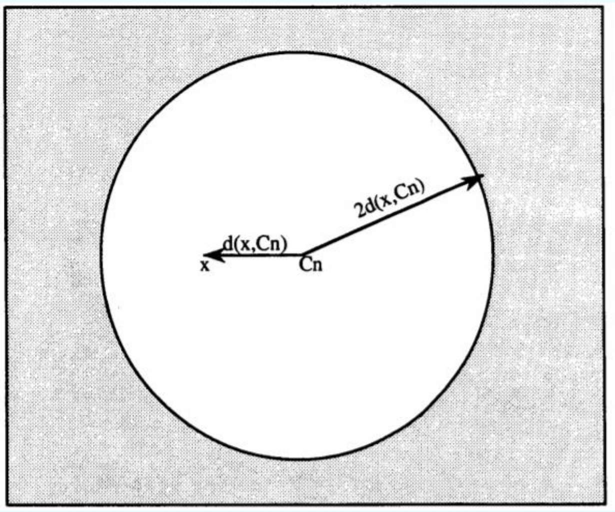
Ainsi, nous posons d'abord Cn premier vecteur du dictionnaire puis le comparons à chacun des vecteurs du dictionnaire Cs, et éliminons les vecteur Cp par l'inégalité triangulaire.

Remarque

On vois ici qu'il est pertinent de choisir un bon Cs pour éliminer un maximum de vecteurs.

Ainsi dès que l'on trouve un Cs plus petit que Cn, nous les inversons car Cn est potentiellement un bon candidat en tant que Cs car ancien vecteur le plus proche de X.

Stratégie d'élimination de vecteur : Inégalité triangulaire deuxième version



Il existe une autre forme plus efficace de l'élimination par inégalité triangulaire que nous utilison aussi dans notre algorithme.

Posons :
***Ci** le vecteur actuellement pris dans le dictionnaire*
***Cj** le vecteur que nous comparons à Cn.*
***X** le vecteur de l'image.*

Si Cj répond à la condition suivante, alors il peut être éliminé.

$$d(c_i , c_j) > 2 \cdot d(x , c_i)$$

Cela peut être compris via l'image ci-dessus qui représente l' inégalité.

Ainsi nous éliminons également les vecteurs avec cette inégalité.

Remarque

En plus de cela, lorsque nous trouvons un nouveau Cn, nous le passons à l'inégalité ci-dessu avec Cj le vecteur le plus proche de Ci.

Si cette inégalité est vraie, alors Cn est forcément le vecteur le plus proche de X.

Stratégie : Choix pertinent du Cn

Comment dit précemement le choix du Cn est important pour éliminer un maximum de vecteur d'un coup par l'inégalité triangulaire.

De plus dans une image, les pixels contigües sont souvent ressemblant voir identique. (Comme on peut le voir sur l'image ci-contre)

Ainsi il apparait pertient lorsque l'on trouve un vecteur du dictionnaire Cn correspond au vecteur Xn de le réutilisé comme Cn pour le prochain vecteur Xn+1.

C'est pour cela que notre algorithme réordonne l'ordre de parcours des vecteurs du dictionnaire en remettant en premier le vecteur Cn trouvé précedemment.

Conclusion

Notre algorithme élimine donc de plus en plus vecteur au cours de la recherche du vecteur du dictionnaire Cn le plus proche du vecteur X de l'image.

Il s'adapte également au résultat trouvé précédemment pour trouver plus rapidemment le prochain plus proche vecteur.

Il se démarque donc d'un algorithme naïf calculant chaque distance pour chaque vecteur du dictionnaire et de l'image.

Référence :

ALAIN NYECK : ETUDE ET MISE AU POINT D'ALGORITHMES RAPIDES DE QUANTIFICATION VECTORIELLE . APPLICATION AU CODAGE D'IMAGES