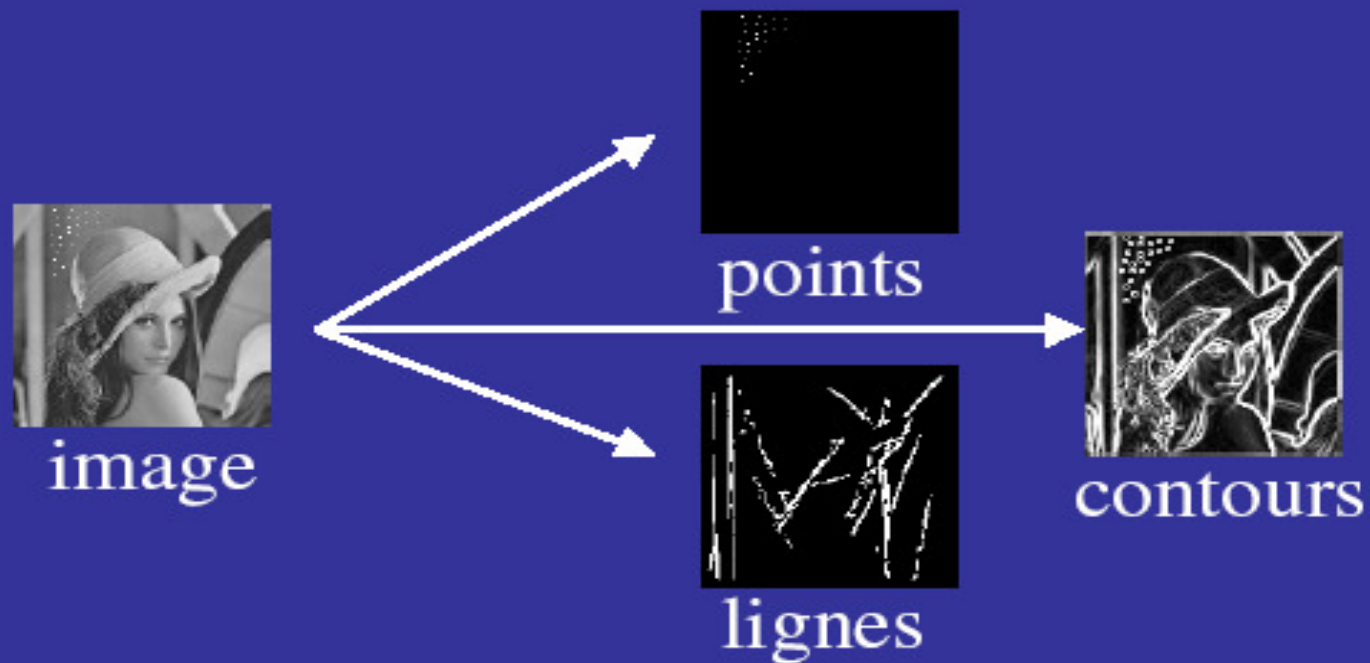
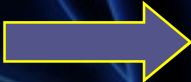


Points d'intérêt et coins.

Éléments caractéristiques



Points d'intérêt et coins.

- On se pose souvent la question de savoir quels sont les endroits de l'image contenant le plus d'information
- Nous nous intéressons maintenant aux coins et aux points d'intérêt. Dès 1977, H. Moravec introduit la notion de points d'intérêt. Pour lui, certains points dans une image **peuvent avoir des caractéristiques plus significatives** que les autres.
- P.R. Beaudet (1978)  Formalisme Mathématique (détecteur)

Détection de points d'intérêt

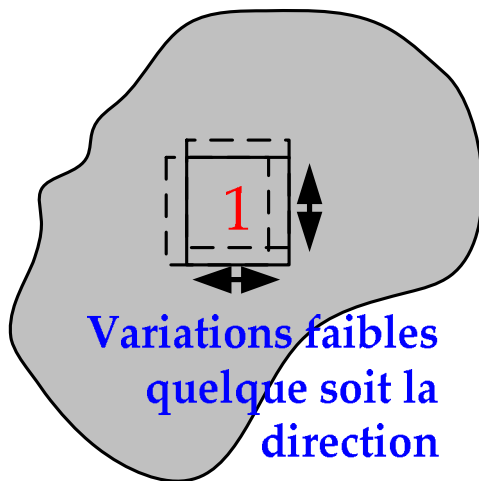
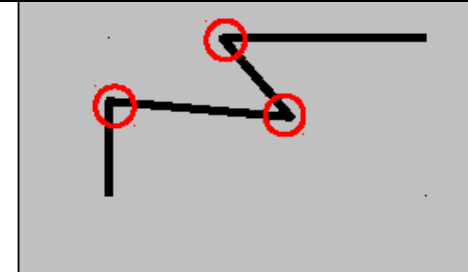
- Il existe plusieurs opérateurs permettant de retrouver des points d'intérêt dans une image:

deux grandes catégories

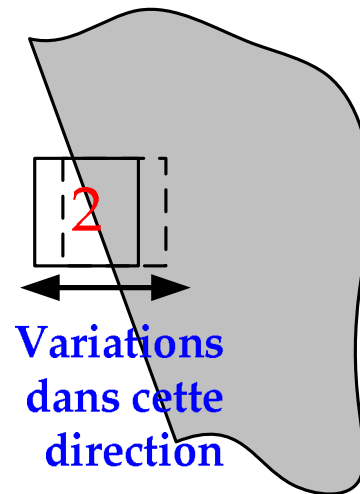
- Après une détection de contours, rechercher les points de courbure maximale (les détecteurs utilisant les contours et leurs fortes courbures)
- les détecteurs utilisant une représentation directe des coins.

Points d'intérêt et coins.

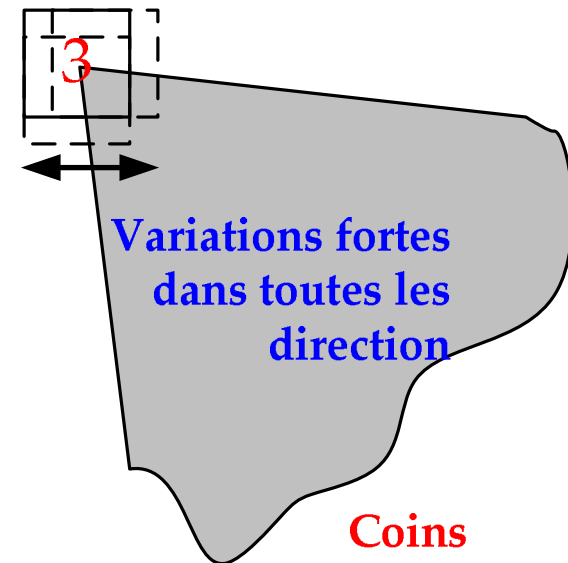
- Un coin est un croisement de la dérivée x avec la dérivée y
- Détecteur de Harris/Stephens (1988)



Zone Homogène



Arête



Coins

Points d'intérêt et coins.

- La détection des points d'intérêt (ou *coins*) est, au même titre que la détection de contours, une **étape préliminaire à de nombreux processus de vision par ordinateur.**
- Les points d'intérêt, dans une image, correspondent à des **doubles discontinuités de la fonction d'intensités.**

Points d'intérêt et coins.

- Les points d'intérêt sont des points où le signal de l'image accuse un changement brusque dans les deux dimensions spatiales. Des exemples sont les coins, les jonctions en T, et aussi les endroits où la texture varie fortement.
- Pour détecter le changement bidimensionnel, plusieurs approches ont été proposées, depuis des années 80.

Points d'intérêt et coins.



Différents types de points d'intérêt :

Coins, jonction en T et point de fortes variations de texture.

Avantages des points d'intérêt.

- Sources d'informations plus fiable que les contours car plus de contraintes sur la fonction d'intensité.

➡ Robuste aux occultations (soit occulté, soit visible).

➡ Pas d'opérations de chaînage (-> contours !)

➡ Présents dans une grande majorité d'images (\neq contours !).

Détecteurs des points d'intérêt.

DETECTEUR DE SUSAN

DETECTEUR DE HARRIS

HARRIS : MODELE ELECTROSTATIQUE

DETECTEUR DE SUSAN

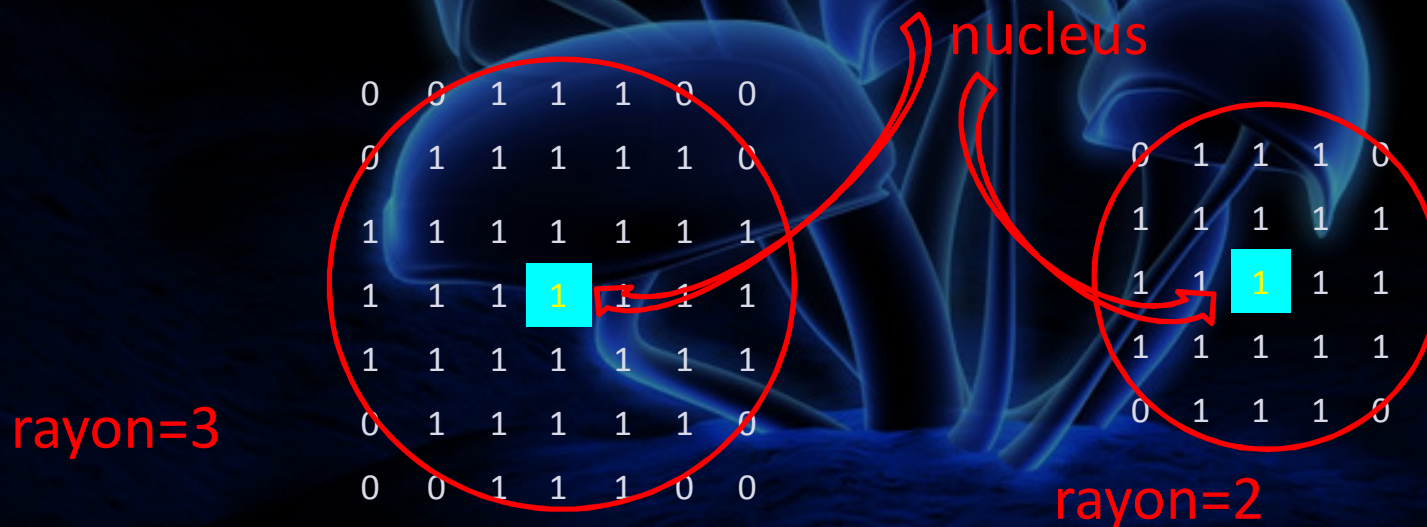
Algorithme développé à l'université d'Oxford.

SUSAN → *Univalue Segment Assimilating Nucleus*



Détecteur de SUSAN

- ❖ Le niveau de gris (NDG) de chaque pixel du disque est comparé avec celui du "nucleus" et on définit alors la zone USAN.
- ❖ La zone "USAN" contient beaucoup d'information locale sur la structure de l'image.



Détecteur de SUSAN

Cette approche de la détection d'objet a l'avantage de ne nécessiter aucune dérivée .

- 👉 La zone "USAN" est de taille maximale lorsque le "nucleus" se trouve dans une région de NDG uniforme.
- 👉 Elle est deux fois moins importante lorsqu'il se trouve près d'un contour.
- 👉 Et est encore plus faible lorsque le "nucleus" tombe dans un coin.

Détecteur de SUSAN

Principe de SUSAN

SUSAN → *Smallest Univalve Segment Assimilating Nucleus*.

Donnant en sortie une matrice de la même taille que l'image qui contient des informations pour:

Détection des contours et des coins.

les coins étant détectés plus fortement que les contours.

Détecteur de SUSAN

Tous les pixels à l'intérieur d'un masque circulaire sont comparés avec le niveau de gris du "nucleus", à l'aide de l'équation de comparaison suivante:

$$c(\vec{r}, \vec{r}_0) = e^{-\left[\frac{I(\vec{r}) - I(\vec{r}_0)}{t}\right]^6}$$

Le seuil t correspond au minimum de contraste de détermination des contours

la taille de la zone USAN (i.e. nombre de pixels de même NDG que le "nucleus") est donnée par:

$$n(\vec{r}_0) = \sum_{\vec{r}} c(\vec{r}, \vec{r}_0)$$

Détecteur de SUSAN

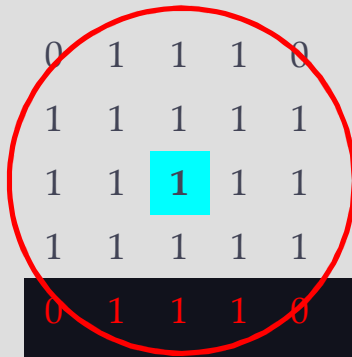
Image →
25x30



Détecteur de SUSAN

Image →
25x30

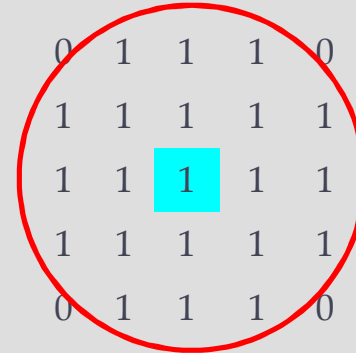
$n=18$



Détecteur de SUSAN

Image →
25x30

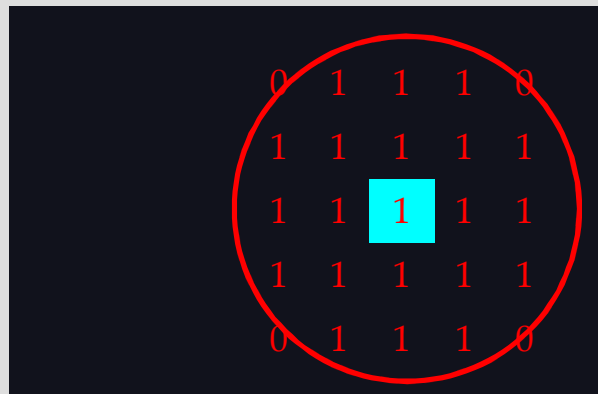
n=21



0	1	1	1	0
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
0	1	1	1	0

Détecteur de SUSAN

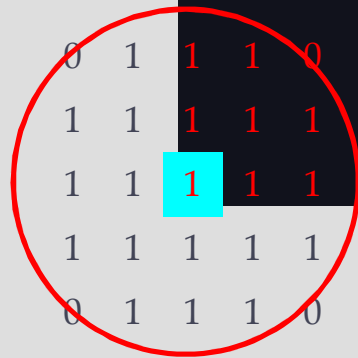
Image →
25x30



$n=21$

Détecteur de SUSAN

Image →
25x30



$n=8$

[illegible][illegible][illegible]

Détecteur de SUSAN

8<=Seuil<11

[illegible]

Détecteur de SUSAN

11<=Seuil<13

[illegible]

Détecteur de SUSAN

13<=Seuil<14

[illegible]

Détecteur de SUSAN

14<=Seuil<15

[illegible]

Détecteur de SUSAN

On remarque que les points détectés dépend d'un seuil

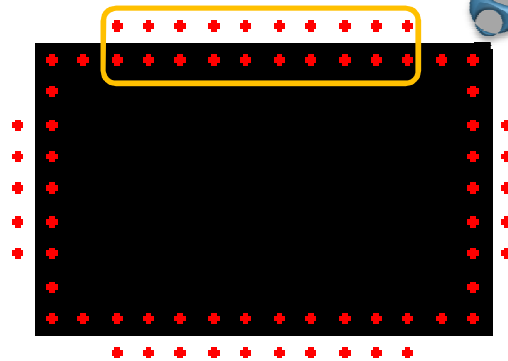
$\text{Seuil} = \min f + \alpha * (\max f - \min f)$ avec : $0 < \alpha < 1$

Détecteur de SUSAN



Détecteur de SUSAN

Plusieurs points
très proches ???



Sélection

Détecteur de SUSAN

On met les résultats dans une matrice **f** de même taille que l'image:

[illegible]

Détecteur de SUSAN

On met les résultats dans une matrice **f** de même taille que l'image:

[illegible]

Détecteur de SUSAN

On met les résultats dans une matrice **f** de même taille que l'image:

[illegible]

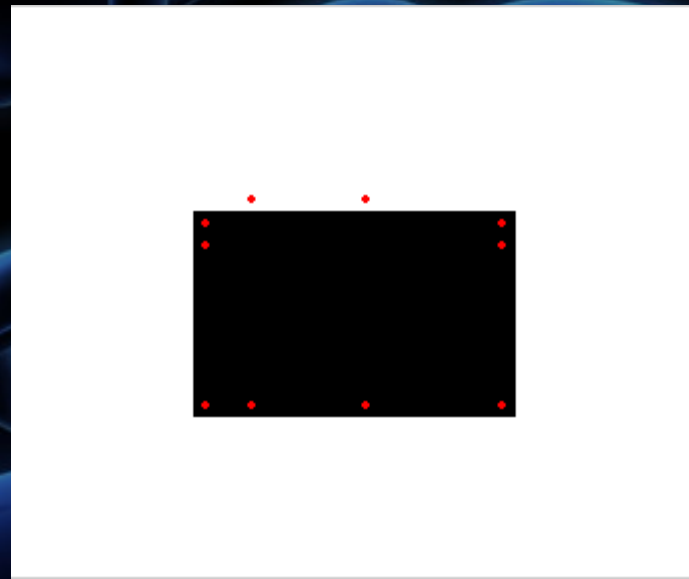
Détecteur de SUSAN

En balayant toute l'image par une fenêtre de 5x5 on trouve les résultats suivants:

[illegible]

Détecteur de SUSAN

Alors on obtient des points relativement séparés entre eux.



Détecteur de SUSAN

Pour la même image mais de taille plus grande que la première:

Seuil=14.5
 $r=10$

Image →
256x318

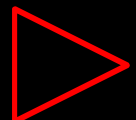


Algorithme de SUSAN

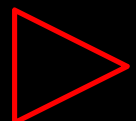
Chargement et conversion de l'image



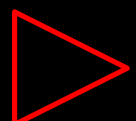
Génération du Masque



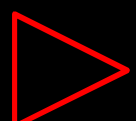
Matrice USAN



Seuillage et sélection des points d'intérêt

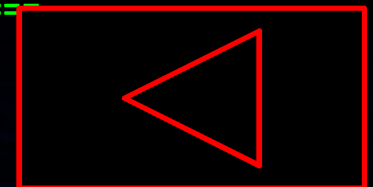


Affichage des résultats



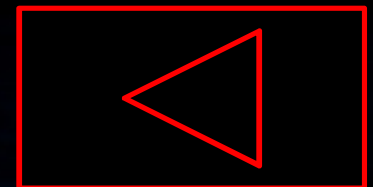
Détecteur de SUSAN

```
clear all
close all
clc
% =====chargement et conversion de l'image=====
im = imread('2rect.bmp');
% =====conversion de l'image=====
d = length(size(im));
% im = imnoise(im,'speckle',0.02);
if d==3
    image=double(rgb2gray(im));
elseif d==2
    image=double(im);
end
subplot(1,2,1)
imshow(im)
[n,m]=size(image);
% =====données=====
rayon=1;
alpha=80;
r=5;
alpha=alpha/100;
```



Détecteur de SUSAN

```
mask=zeros(2*rayon+1);  
b=ones(rayon+1);  
for i=1:rayon+1  
    for j=1:rayon+1  
        if (rayon==1)  
            if(j>i)  
                b(i,j)=0;  
            end  
        else  
            if(j>i+1)  
                b(i,j)=0;  
            end  
        end  
    end  
end  
mask(1:rayon+1,rayon+1:2*rayon+1)=b;  
mask(1:rayon+1,1:rayon+1)=rot90(b);  
mask0=mask;  
mask0=flipdim(mask0,1);  
mask=mask0+mask;  
mask(rayon+1,:)=mask(rayon+1
```



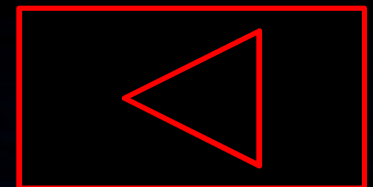
Détecteur de SUSAN

```
% =====réponse maximale=====
max_reponse=sum(sum(mask));
% =====balayage de toute l'image=====
f=zeros(n,m);
for i=(rayon+1):n-rayon
    for j=(rayon+1):m-rayon

        image_courant=image(i-rayon:i+rayon,j-rayon:j+rayon);

        image_courant_mask=image_courant.*mask;

        inteniste_cental= image_courant_mask(rayon+1,rayon+1);
        s=exp(-1*(((image_courant_mask-inteniste_cental)/max_reponse).^6));
        somme=sum(sum(s));
% si le centre du mask est un 0 il faut soustraire les zeros des filtres
        if (inteniste_cental==0)
            somme=somme-length((find(mask==0)));
        end
        f(i,j)=somme;
    end
end
```

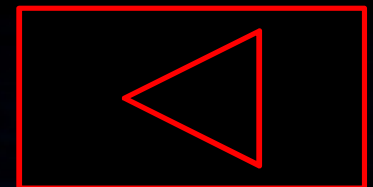


Détecteur de SUSAN

```
% =====selection et seuillage des points d'interêt=====
ff=f(rayon+1:n-(rayon+1),rayon+1:m-(rayon+1));
minf=min(min(ff));
maxf=max(max(f));
fff=f;
d=2*r+1;
temp1=round(n/d);
if (temp1-n/d)<0.5 &(temp1-n/d)>0
    temp1=temp1-1;
end
temp2=round(m/d);
if (temp2-m/d)<0.5 &(temp2-m/d)>0
    temp2=temp2-1;
end
fff(n:temp1*d+d,m:temp2*d+d)=0;
```

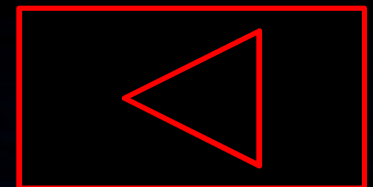
Détecteur de SUSAN

```
for i=(r+1):d:temp1*d+d
for j=(r+1):d:temp2*d+d
    window=fff(i-r:i+r,j-r:j+r);
    window0=window;
    [xx,yy]=find(window0==0);
    for k=1:length(xx)
        window0(xx(k),yy(k))=max(max(window0));
    end
    minwindow=min(min(window0));
[y,x]=find(minwindow~=window & window<=minf+alpha*(maxf-minf) & window>0);
[u,v]=find(minwindow==window);
if length(u)>1
    for l=2:length(u)
        fff(i-r-1+u(l),j-r-1+v(l))=0 ;
    end
end
if length(x)~=0
    for l=1:length(y)
        fff(i-r-1+y(l),j-r-1+x(l))=0 ;
    end
end
end
end
end
seuil=minf+alpha*(maxf-minf);
[u,v]=find(minf<=fff & fff<=seuil );
```



Détecteur de SUSAN

```
% =====affichage des resultats=====
subplot(1,2,2)
imshow(im)
hold on
plot(v,u,'.r','MarkerSize',10)
nombre_de_point_dinteret=length(v)
```





DETECTEUR DE SUSAN

DETECTEUR DE HARRIS

HARRIS : MODELE ELECTROSTATIQUE

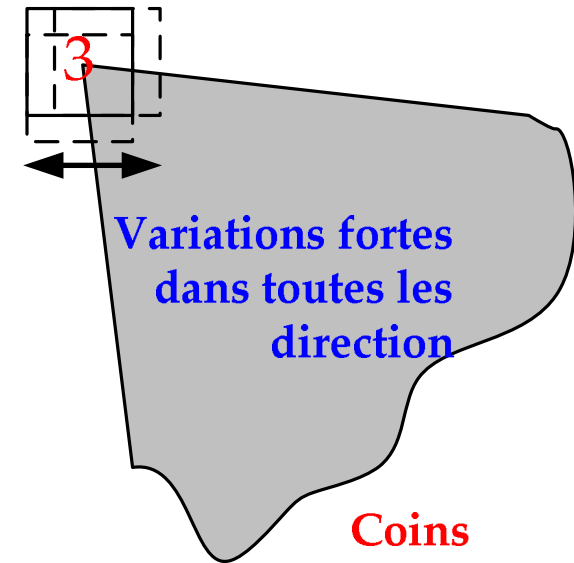
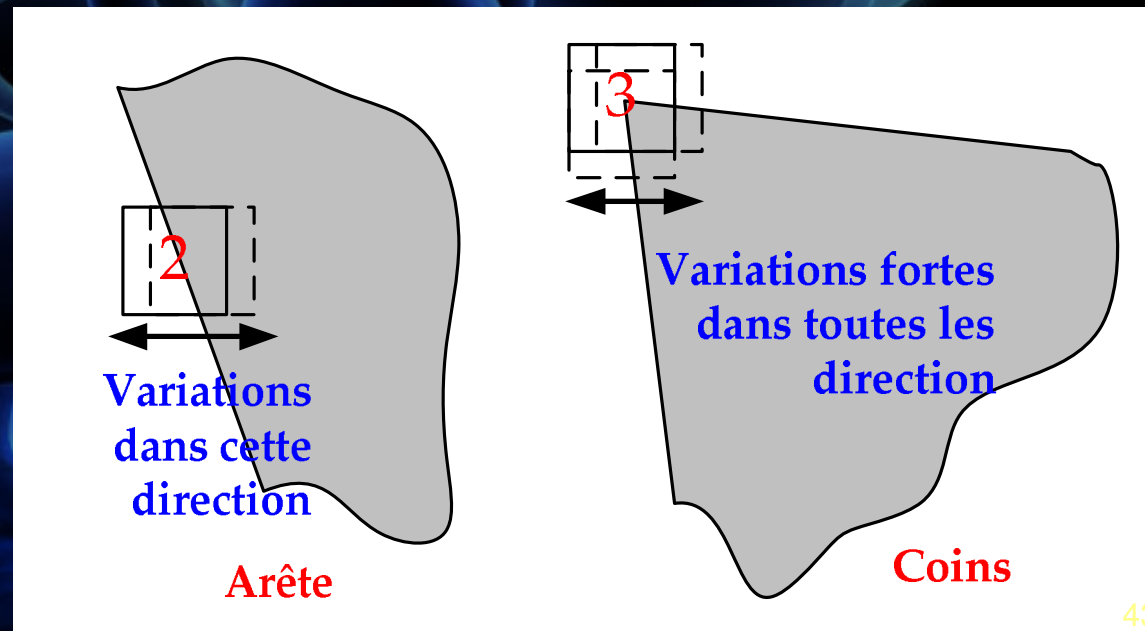
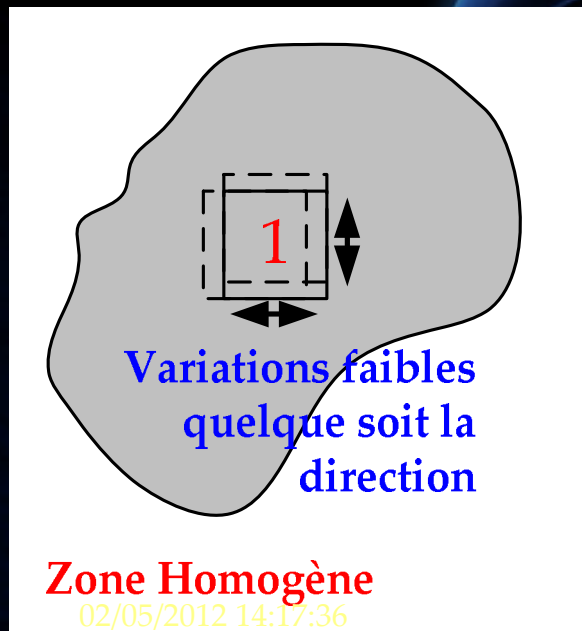
Détecteur de Harris (Harris, 1998)

- Elle se fonde sur l'analyse de la variation de la luminance au voisinage d'un point (Pixel) : l'approche intensité.
- Mesurer le changement des niveaux de gris d'une fenêtre, centrée autour d'un point, quand elle est déplacée dans différentes directions.

Détecteur de Harris :

Trois situations se présentent :

- Faibles variations dans toutes les directions → Zone homogène.
- Fortes variations dans une direction prédominante → Arête.
- Fortes variations quelque soit la direction → Coins.



Détecteur de Harris :

Les valeurs propres λ_1, λ_2 de la matrice M représentent les courbures principales de la fonction d'auto-corrélation, nous avons alors trois cas, si :

- Les deux valeurs propres sont faibles, la région considérée est homogène.
- Une seule des valeurs propres est clairement dominante, on se trouve sur une arête.
- Les deux valeurs propres sont élevées, on se trouve en présence d'un point d'intérêt : coin.

Détecteur de Harris :

- Le problème est donc l'évaluation de ces valeurs propres.
- Eviter de faire le calcul des valeurs propres, qui est relativement complexe.
- Harris et Stephens proposent de calculer une mesure s'appuyant sur le déterminant et la trace de la matrice M .

Détecteur de coins de Harris :

➤ On évalue alors la mesure suivante: Fonction de Harris

$$R = \det(M) - k \cdot \text{Trace}^2(M)$$

Avec : $\det(M) = \lambda_1 \cdot \lambda_2 = \langle I_x^2 \rangle \langle I_y^2 \rangle - \langle I_x I_y \rangle^2$

$$\text{Trace}(M) = \lambda_1 + \lambda_2 = \langle I_x^2 \rangle + \langle I_y^2 \rangle$$

k : paramètre de Harris ($0.04 < k < 0.06$)

➤ L'expérimentation montre selon plusieurs auteurs qu'un nombre optimal de points est obtenu pour une valeur de l'ordre de $k=0.04$.

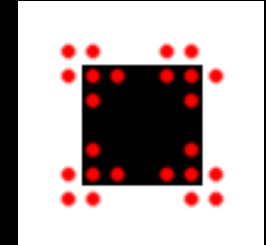
Détecteur de coins de Harris :

- $R < 0$ au voisinage d'une arête.
- $R = 0$ dans une région homogène.
- $R > 0$ au voisinage d'un point d'intérêt (coin).

Détecteur de coins de Harris :

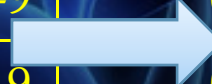
Seuillage :

➤ Eliminer les valeurs qui sont inférieure à un seuil modifiable de la réponse R.



seuil=40

24	34	12	-4	-9	-9	-4	12	34	25
34	71	64	19	-9	-9	19	64	72	34
12	64	99	52	2	2	52	99	64	12
-4	19	52	33	5	5	33	52	19	-4
-9	-9	2	5	1	1	5	2	-9	-9
-9	-9	2	5	1	1	5	2	-9	-9
-4	19	52	33	5	5	33	52	19	-4
12	64	99	52	2	2	52	99	64	12
34	72	64	19	-9	-9	19	64	72	34
25	34	12	-4	-9	-9	-4	12	34	25

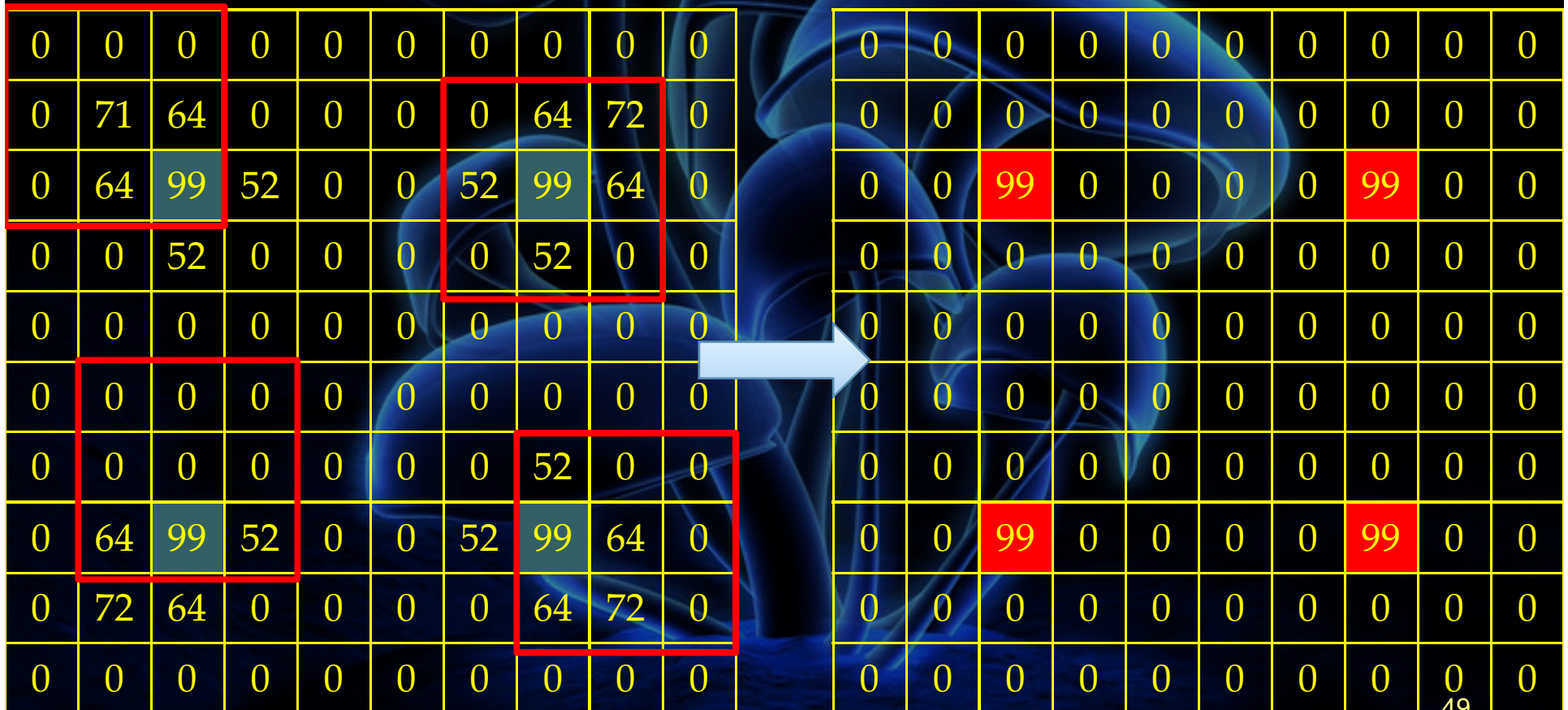
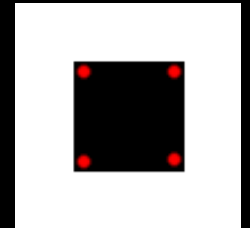


0	0	0	0	0	0	0	0	0	0
0	71	64	0	0	0	0	64	72	0
0	64	99	52	0	0	52	99	64	0
0	0	52	0	0	0	0	52	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	52	0	0	0	0	52	0	0
0	64	99	52	0	0	52	99	64	0
0	72	64	0	0	0	0	64	72	0
0	0	0	0	0	0	0	0	0	0

Détecteur de coins de Harris :

Extraction des maxima locaux :

➤ A partir de la nouvelle réponse R, on fait le balayage de R avec une fenêtre pour extraire les maxima locaux .



Détecteur de Harris

- Calcul en chaque point de $\frac{\partial I}{\partial x}$ et $\frac{\partial I}{\partial y}$
- Création d'une matrice
 $M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}$ avec : $A = \frac{\partial^2 I}{\partial x^2}$ $B = \frac{\partial^2 I}{\partial y^2}$ $C = \frac{\partial^2 I}{\partial x \partial y}$
- Lissage par un filtre Gaussien ou moyenneur
- Résultat : $H = \text{Det}(M) - k \times \text{Trace}(M)^2$
avec : $\text{Det}(M) = AB - C^2$ $\text{Trace}(M) = A + B$ $0 \leq k \leq 0.25$
- Comparaison à un seuil

Détecteur de Harris : Algorithme 1

- Calculer les images des gradients X et Y
- Filtrer l'image par un filtre Gaussien
- Pour chaque pixel de l'image :
 - Calculer la matrice M sur une fenêtre de taille définie
 - Calculer les valeurs propres λ_1, λ_2 de la matrice M
 - Si les deux valeurs propres sont élevées (seuil) : détection d'un coin
- Conserver les maximum locaux pour les coins détectés

$$M = \begin{bmatrix} a & b \\ b & d \end{bmatrix}$$

$$\lambda_1 = \frac{1}{2} \left(a + d - \sqrt{a^2 + 4b^2 - 2ad + d^2} \right)$$

$$\lambda_2 = \frac{1}{2} \left(a + d + \sqrt{a^2 + 4b^2 - 2ad + d^2} \right)$$

*On ne calcule pas
comme cela en
pratique*

Détecteur de Harris : Algorithme 2

- Plutôt que de calculer les valeurs propres, il est possible de calculer les valeurs suivantes :
 - $\text{Trace}(M) = \lambda_1 + \lambda_2 = M_{1,1} + M_{2,2}$
 - $\text{Déterminant}(M) = \lambda_1 \lambda_2 = M_{1,1}M_{2,2} - M_{1,2}M_{2,1}$
- Et on calcule la réponse suivante :
 - $R = \text{Déterminant}(M) - k (\text{Trace}(M))^2$
 - Les coins correspondent aux maximums locaux de R où
 - $R > 0$
 - $\text{Trace}(M) > t$
 - Constante $k = 0.04$ (typiquement)
 - Constante t selon le nombre de points voulus

❖ Code MATLAB :

```
img=imread('lena.png');
imd=double(img);
sigma=1; k=0.04; w=5*sigma; seuil=50; r=6; size=2*r+1;
dx=[-1 0 1]; dy=dx'; % filtre dérivatif

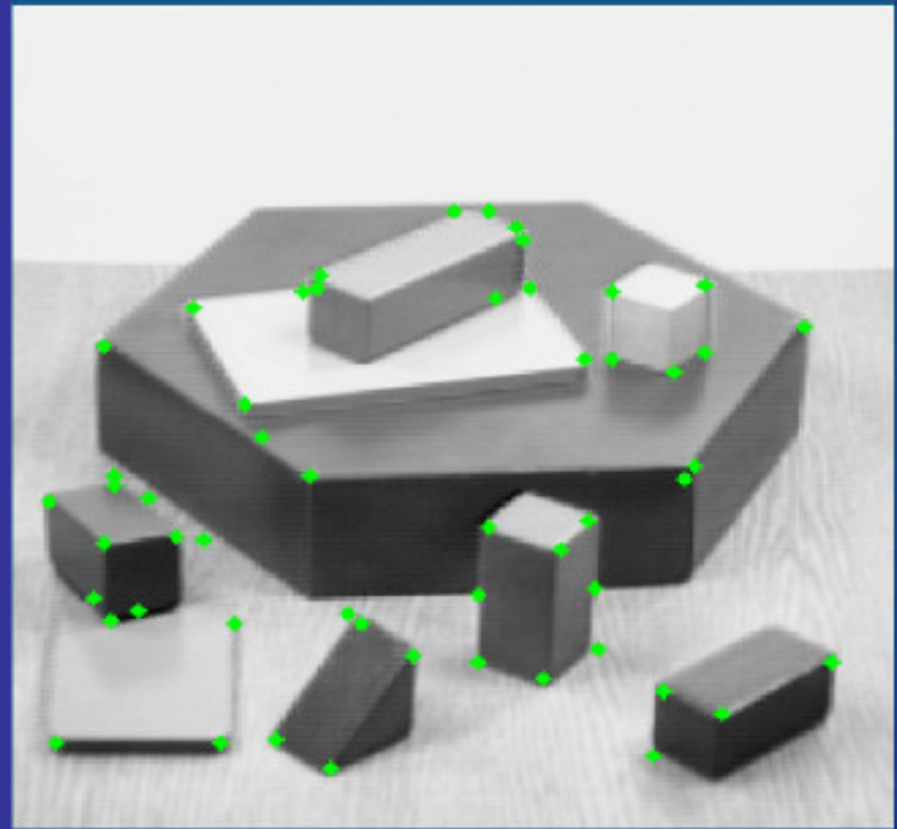
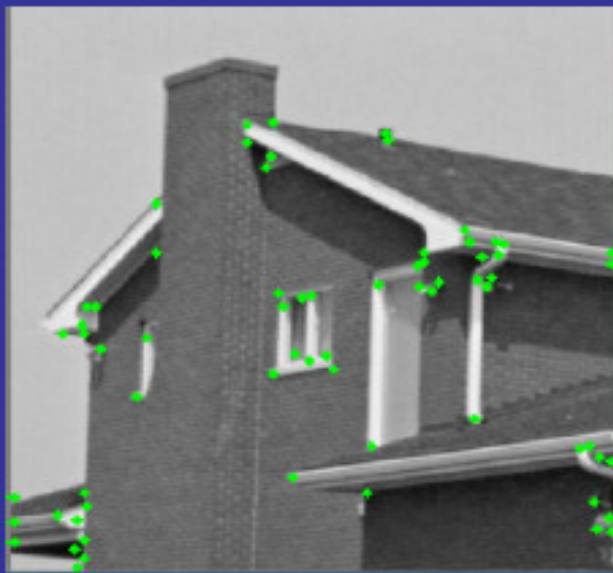
g = fspecial('gaussian',max(1,fix(w)), sigma); % filtre gaussien
Ix=conv2(imd,dx,'same');
Iy=conv2(imd,dy,'same');
Ix2 = conv2(Ix.^2, g, 'same');
Iy2 = conv2(Iy.^2, g, 'same');
Ixy = conv2(Ix.*Iy, g, 'same');
R=Ix2.*Iy2-Ixy.^2-k*(Ix2+Iy2).^2;
R1=(1000/(max(max(R))))*R;
%***** Seuillage et extraction des points d'intérêt *****
[u,v]=find(R1<=seuil);
nb=length(u);
for l=1:nb
    R1(u(l),v(l))=0;
end
R11=zeros(m+2*r,n+2*r);
R11(r+1:m+r,r+1:n+r)=R1;
```

❖ Code MATLAB :

```
[m1,n1]=size(R11);
for i=r+1:m1-r
    for j=r+1:n1-r
        fenetre=R11(i-r:i+r,j-r:j+r);
        ma=max(max(fenetre));
        if fenetre(r+1,r+1)<ma
            R11(i,j)=0;
        end
    end
end
R11=R11(r+1:m+r,r+1:n+r);
[x,y]=find(R11);
%=====
MX=ordfilt2(R1,size^2,ones(size));
R11 = (R1==MX)&(R1>seuil);
[x,y]=find(R11);
%***** affichage des points d'intérêt *****
nb=length(x);
imshow(img)
hold on
plot(y,x,'r')
```


Détecteur de Harris

Quelques exemples





DETECTEUR DE SUSAN

DETECTEUR DE HARRIS

HARRIS : MODELE ELECTROSTATIQUE

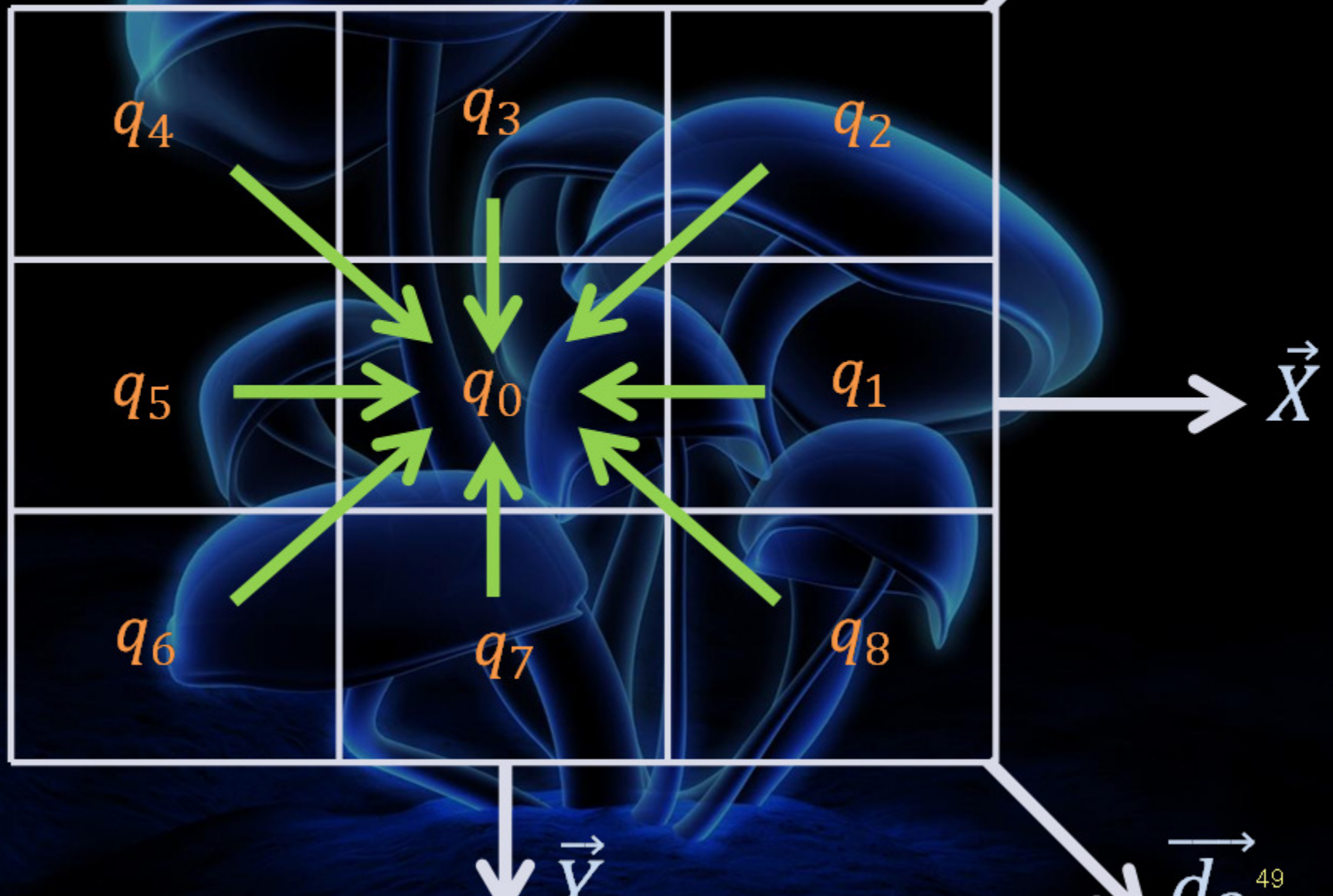
Modèle électrostatique

D. Aboutajdine, et al 2, GVIP :September, 2006)

- C'est une nouvelle méthode de détection des points d'intérêts basée sur l'algorithme de Harris.
- L'idée de base est de modéliser l'image sous forme d'une répartition de charges électriques dans un plan surfacique.
- Le niveau de gris de l'image peut se modéliser sous forme d'un ensemble de points des charges électriques uniformément distribuées sur un plan surfacique, dans la balance électrostatique.

Modèle électrostatique :

Bloc de taille 3*3



Modèle électrostatique :

➤ Les forces horizontales et verticales (forces attractives):

$$\vec{F}_h = \vec{F}(q_1/q_0) + \vec{F}(q_5/q_0) = \frac{Kq_0}{d^2} (q_5 - q_1) \vec{X}$$

$$\vec{F}_v = \vec{F}(q_3/q_0) + \vec{F}(q_7/q_0) = \frac{Kq_0}{d^2} (q_3 - q_7) \vec{Y}$$

Avec: $K = \frac{1}{4\pi\epsilon_0} = \text{Constante}$

$d = \text{distance entre } q_i \text{ et } q_0$

Modèle électrostatique :

➤ Les forces diagonales sont données par :

$$\overrightarrow{F_{d_1}} = \vec{F}(q_6/q_0) + \vec{F}(q_2/q_0) = \frac{Kq_0\sqrt{2}}{4d^2}(q_6 - q_2)(\vec{X} - \vec{Y})$$

$$\overrightarrow{F_{d_2}} = \vec{F}(q_4/q_0) + \vec{F}(q_8/q_0) = \frac{Kq_0\sqrt{2}}{4d^2}(q_4 - q_8)(\vec{X} + \vec{Y})$$

Modèle électrostatique :

➤ La projection de la résultante des forces suivant \vec{X} donne :

$$F_X \vec{X} = \frac{Kp(i,j)}{d^2} \left(q_5 - q_1 + \frac{\sqrt{2}}{4} (q_6 - q_2 + q_4 - q_8) \right) \vec{X}$$

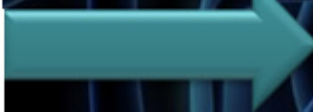
➤ La projection suivant \vec{Y} donne :

$$F_Y \vec{Y} = \frac{Kp(i,j)}{d^2} \left(q_3 - q_7 + \frac{\sqrt{2}}{4} (q_2 + q_4 - q_6 - q_8) \right) \vec{Y}$$

Modèle électrostatique :

➤ Force attractive (q_i et q_0 ont un signe opposé):

Filtre Horizontal



$$dxa = \begin{pmatrix} \frac{\sqrt{2}}{4} & 0 & -\frac{\sqrt{2}}{4} \\ 1 & 0 & -1 \\ \frac{\sqrt{2}}{4} & 0 & -\frac{\sqrt{2}}{4} \end{pmatrix}$$

Filtre Vertical

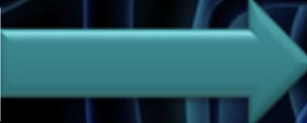


$$dya = \begin{pmatrix} \frac{\sqrt{2}}{4} & 1 & \frac{\sqrt{2}}{4} \\ 0 & 0 & 0 \\ -\frac{\sqrt{2}}{4} & -1 & -\frac{\sqrt{2}}{4} \end{pmatrix}$$

Modèle électrostatique :

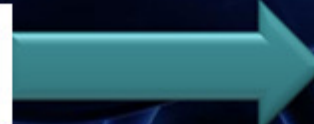
➤ Forces répulsives (q_i et q_0 ont le même signe):

Filtre Horizontal



$$dxr = \begin{pmatrix} -\frac{\sqrt{2}}{4} & 0 & \frac{\sqrt{2}}{4} \\ -1 & 0 & 1 \\ -\frac{\sqrt{2}}{4} & 0 & \frac{\sqrt{2}}{4} \end{pmatrix}$$

Filtre Vertical



$$dyr = \begin{pmatrix} -\frac{\sqrt{2}}{4} & -1 & -\frac{\sqrt{2}}{4} \\ 0 & 0 & 0 \\ \frac{\sqrt{2}}{4} & 1 & \frac{\sqrt{2}}{4} \end{pmatrix}$$

Modèle électrostatique :

➤ Calculer la réponse de détection, en utilisant la relation de Harris déjà citée:

$$R = \det(M) - k \cdot \text{Trace}^2(M)$$

➤ Chercher les maxima locaux, en triant les coins les plus claires, en enlevant les non maxima locaux.

□ Code MATLAB

```
img= checkerboard(21,3,3);
k=0.04; sigma=1; seuil=100; r=6; w=5*sigma;
[m,n]=size(img); imd=double(img);
dxa=[-sqrt(2)/4 0 sqrt(2)/4 ; -1 0 1 ; -sqrt(2)/4 0 sqrt(2)/4];
% dxa=[sqrt(2)/4 0 -sqrt(2)/4; 1 0 -1; sqrt(2)/4 0 -sqrt(2)/4];
dya=dxa'; % dérivée verticale
g=fspecial('gaussian',max(1,fix(5*sigma)),sigma); % gaussien
Ixa=conv2(imd,dxa,'same');
Iya=conv2(imd,dya,'same');
Ixa2 = conv2(Ixa.^2, g, 'same');
Iya2 = conv2(Iya.^2, g, 'same');
Ixya = conv2(Ixa.*Iya, g,'same');
R=Ixa2.*Iya2-Ixya.^2-k*(Ixa2+Iya2).^2;
R1=(1000/(max(max(R))))*R; %normalisation
[u,v]=find(R1<=seuil);
nb=length(u);
for k=1:nb
    R1(u(k),v(k))=0;
end
```


□ Code MATLAB

```
R11=zeros(m+2*r,n+2*r);
R11(r+1:m+r,r+1:n+r)=R1;
[m1,n1]=size(R11);
for i=r+1:m1-r
    for j=r+1:n1-r
        fenetre=R11(i-r:i+r,j-r:j+r);
        ma=max(max(fenetre));
        if fenetre(r+1,r+1)<ma
            R11(i,j)=0;
        end
    end
end
subplot(1,2,2); imshow(img)
hold on
R11=R11(r+1:m+r,r+1:n+r);
[x,y]=find(R11);
nb=length(x)
plot(y,x,'.r')
title('detection des points d''interet')
```

Modèle électrostatique

❖ Conclusion:

- La méthode électrostatique, donne une bonne performance et les résultats obtenus sont relativement satisfaisantes, dans la comparaison avec ceux qui sont obtenus par l'algorithme de Harris.
- La possibilité d'utiliser cette méthode, pour détecter les points d'intérêt d'une image couleur.
- On peut appliquer cette méthode dans les différents contextes de traitement d'images, et dans la vision par ordinateur ,pour la détection des contours, ainsi que dans la segmentation de l'image.