

# TP n°6

## Modélisation de la distribution du bruit contenu dans une image

### Partie 1 :

Image initiale (1\_Aerial.pgm) :

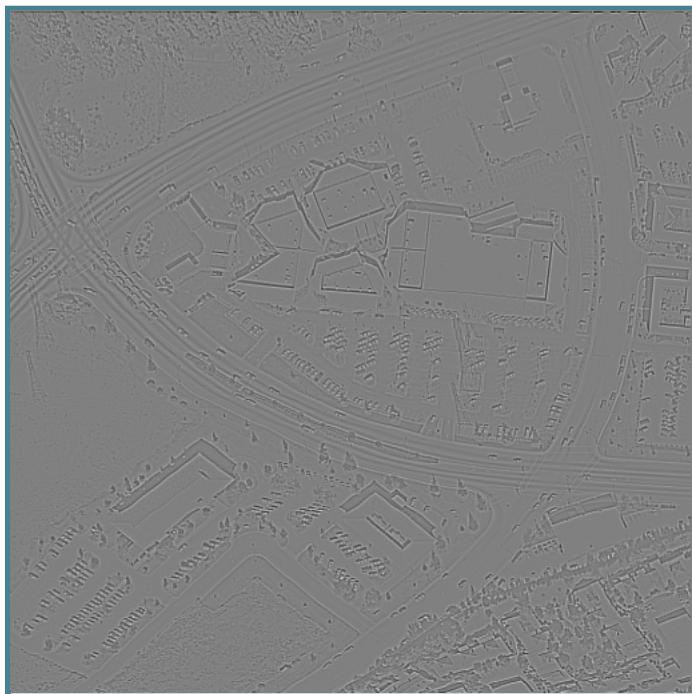


Floutage de l'image (par le programme blur.cpp) :



## Partie 2 :

Bruit obtenu (différence image originale et image floutée) :



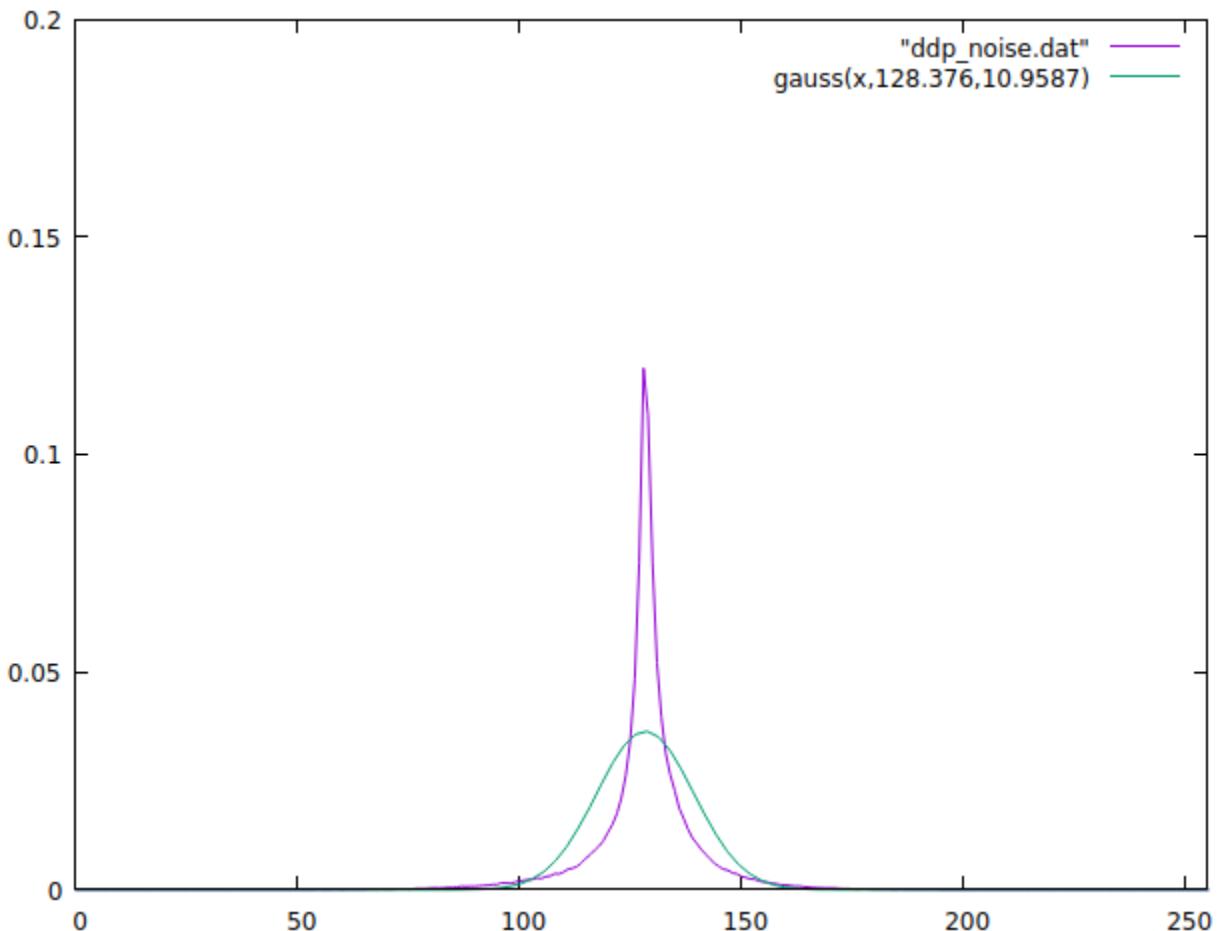
MOYENNE : 128.376

VARIANCE : 120.093

ECART TYPE : 10.9587

En utilisant la fonction de gaussienne dans gnuplot :

```
> gauss(x,mu,sigma)=1/(sigma*sqrt(2.*pi))*exp(-(x-mu)**2./(2.*sigma**2))
```



On se rend compte ici que l'approximation de la DDP du bruit par une gaussienne n'est pas optimale. En effet, avec la gaussienne, le pic est bien présent mais ne colle pas parfaitement à la DDP (gaussienne ou mélange de gaussienne fonctionne mieux pour des images normales).

On pourrait alors se demander comment approximer ce résultat de manière plus exacte, en utilisant un autre type de courbe, comme une Laplacienne.

## Patie 3 :

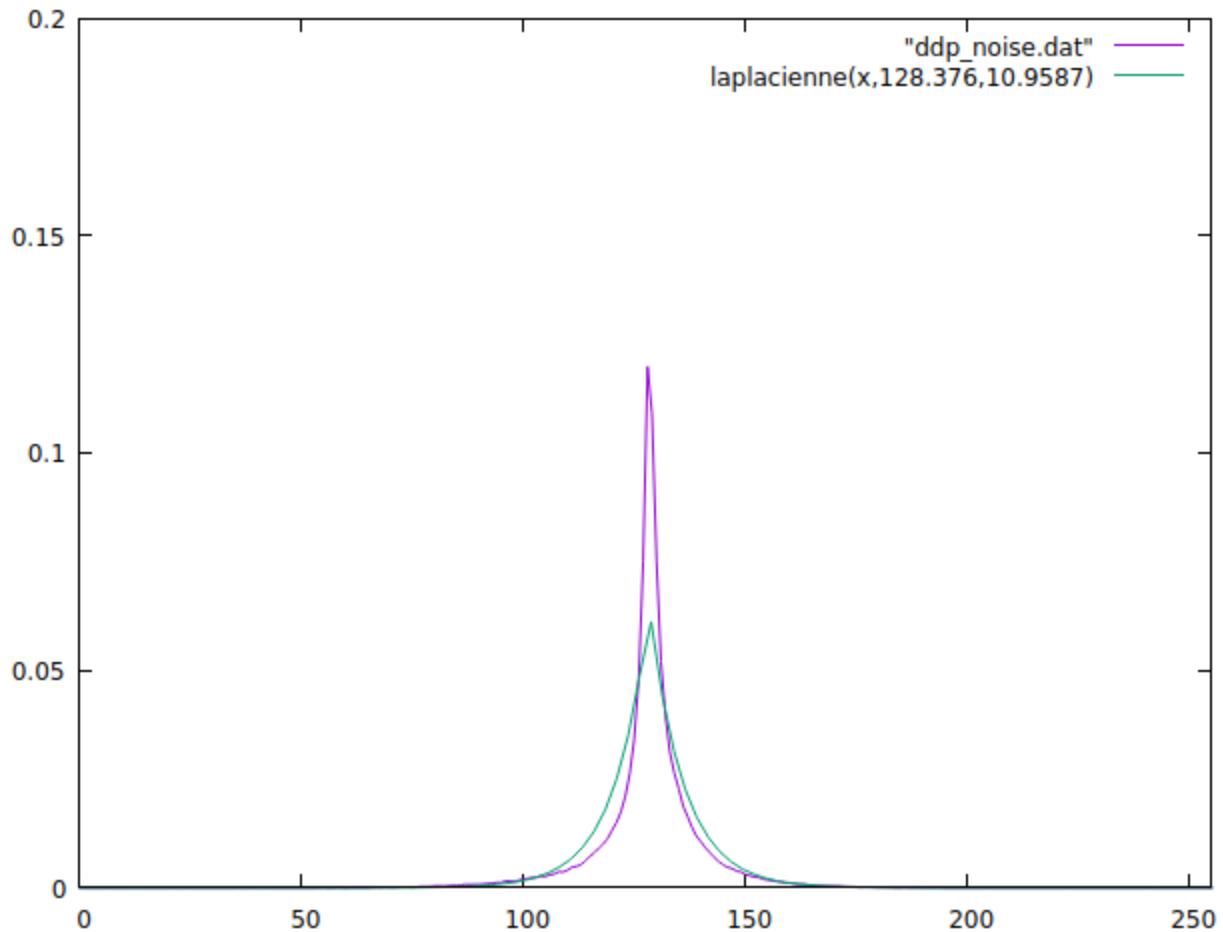
Laplacienne :

D'après la définition de l'écart type d'une Laplacienne, on peut en déduire la valeur de b :

$$\text{Ecarttype} = \text{racinecarrée}(2) * b \Leftrightarrow b = \text{Ecarttype}/\text{racinecarrée}(2)$$

En utilisant la fonction de laplacienne dans gnuplot :

```
> laplacienne(x,mu,sigma)=1./(2.*sigma/sqrt(2)) * exp(-abs(x-mu)/(sigma/sqrt(2)))
```



L'approche du bruit pas un Laplacienne fonctionne mieux (pic plus marqué à la moyenne et moins étendu).

### Partitionnement :

On peut ensuite découper l'image en une série de cadrants de dimensions identiques, afin de partitionner l'image originale en une série de zones analysables indépendamment.

En partitionnant l'image initiale en une série de quatres cadrants dont on va calculer, pour chacun d'eux, une moyenne et une variance/écart type, on obtient le résultat suivant :

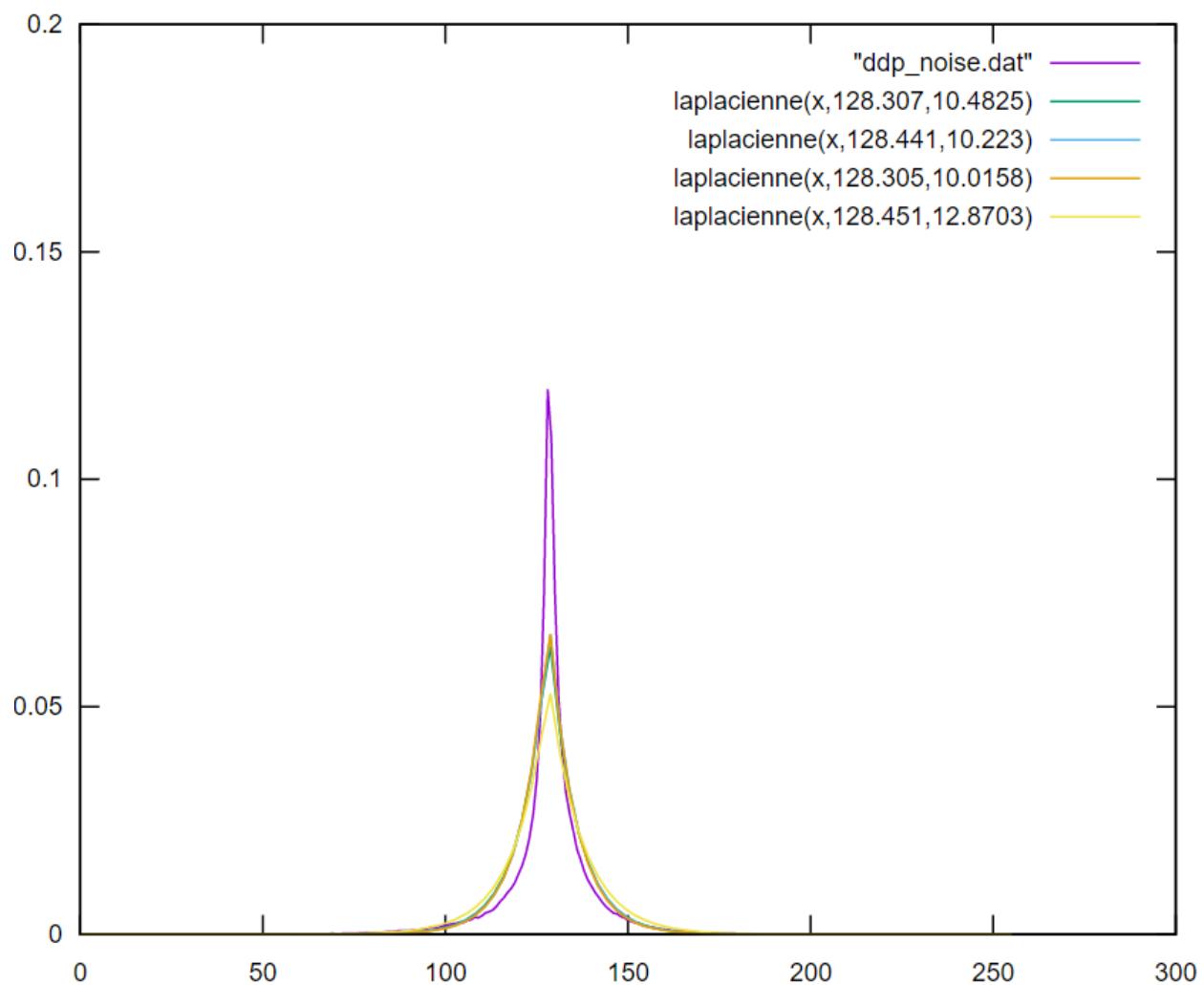
```
===== POUR LE CADRAN : x[0:256], y[0:256] =====
MOYENNE : 128.307
VARIANCE : 109.883
ECART TYPE : 10.4825
===== POUR LE CADRAN : x[0:256], y[256:512] =====
MOYENNE : 128.441
VARIANCE : 104.509
```

```

ECART TYPE : 10.223
===== POUR LE CADRAN : x[256:512], y[0:256] =====
MOYENNE : 128.305
VARIANCE : 100.316
ECART TYPE : 10.0158
===== POUR LE CADRAN : x[256:512], y[256:512] =====
MOYENNE : 128.451
VARIANCE : 165.645
ECART TYPE : 12.8703

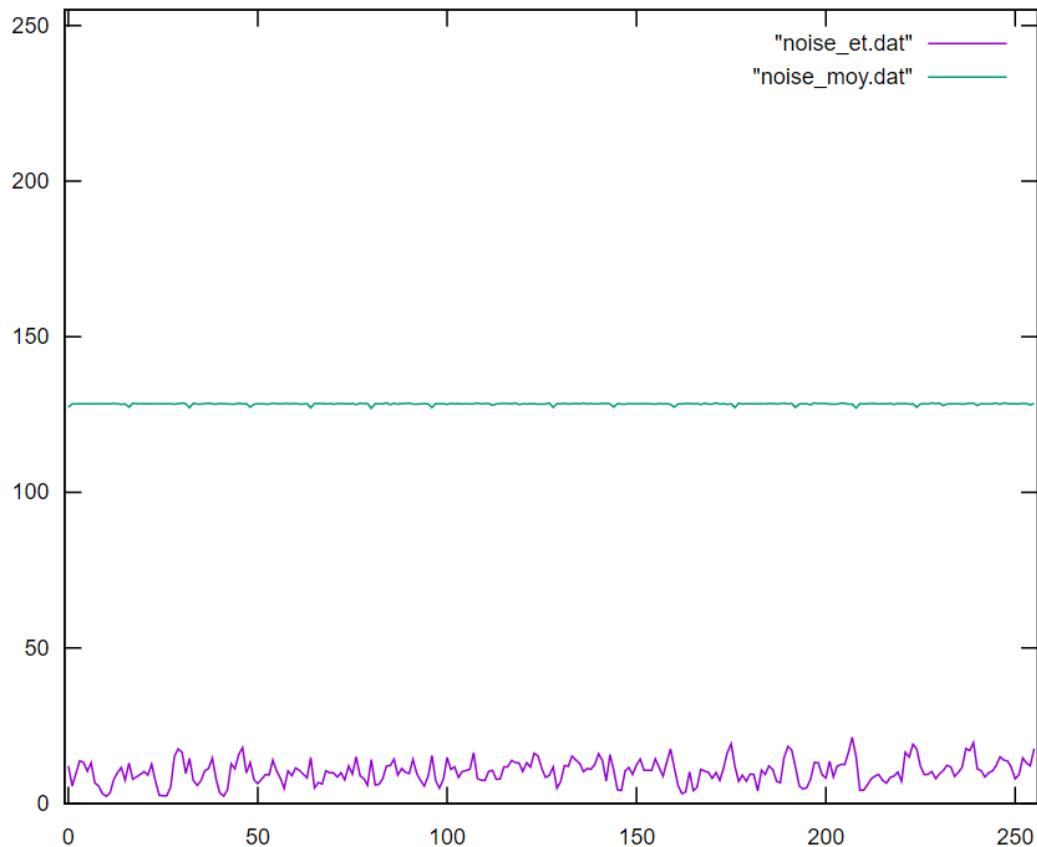
```

On peut ensuite tracer chacune de ces fonctions Laplaciennes sur gnuplot comparativement à la fonction de bruit calculée auparavant:



On constate ici, que les quatres pics se superposent quasiment, car la composition du bruit est très proche d'un cadran à l'autre (premier indicateur de potentielle non falsification).

On peut ensuite faire un partitionnement plus précis en découplant désormais l'image de base à l'aide de 256 cadrans (16x16). On obtient alors en indiquant pour chacun d'eux leur moyenne (courbe verte) et leur écart type (courbe rose), les courbes suivantes dans gnuplot (cadrans en abscisse) :



On se rend compte que la courbe représentant la moyenne de chacun des 256 cadrans est approximable par une courbe constante en 128.

La courbe représentant les écarts types est plus complexe, mais reste bornable en 0-10.

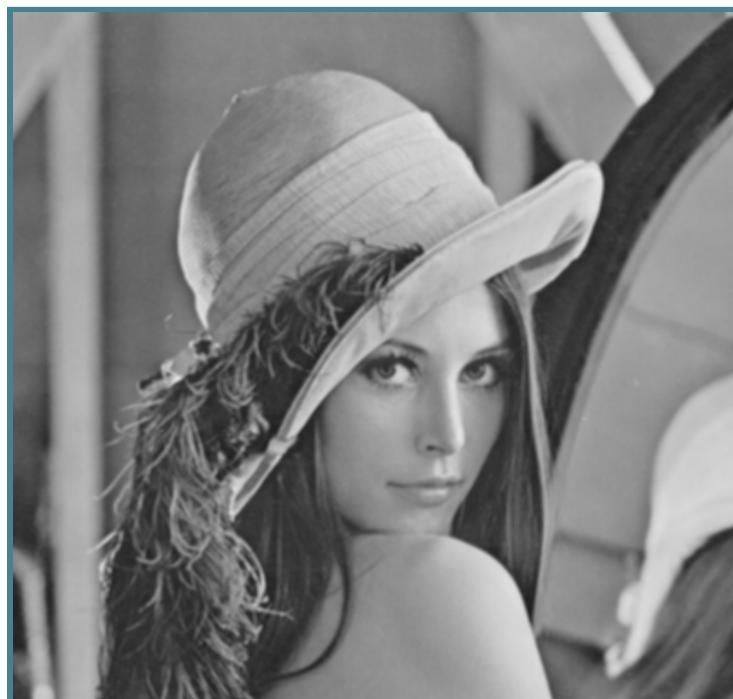
On constate donc que d'un cadran à un autre, la moyenne ne varie pas et l'écart type que faiblement, pour une image non retouchée dont on a extrait le bruit (le bruit est globalement le même pour l'ensemble de l'image).

De plus, la fréquence de répétition de pics similaires dans la courbe des écarts types est constante. En effet, on repère par endroit la répétition de motifs réguliers, sans trop de grandes variations notables.

On peut réitérer le processus avec une autre image plus complexe, comme l'image de Léna :



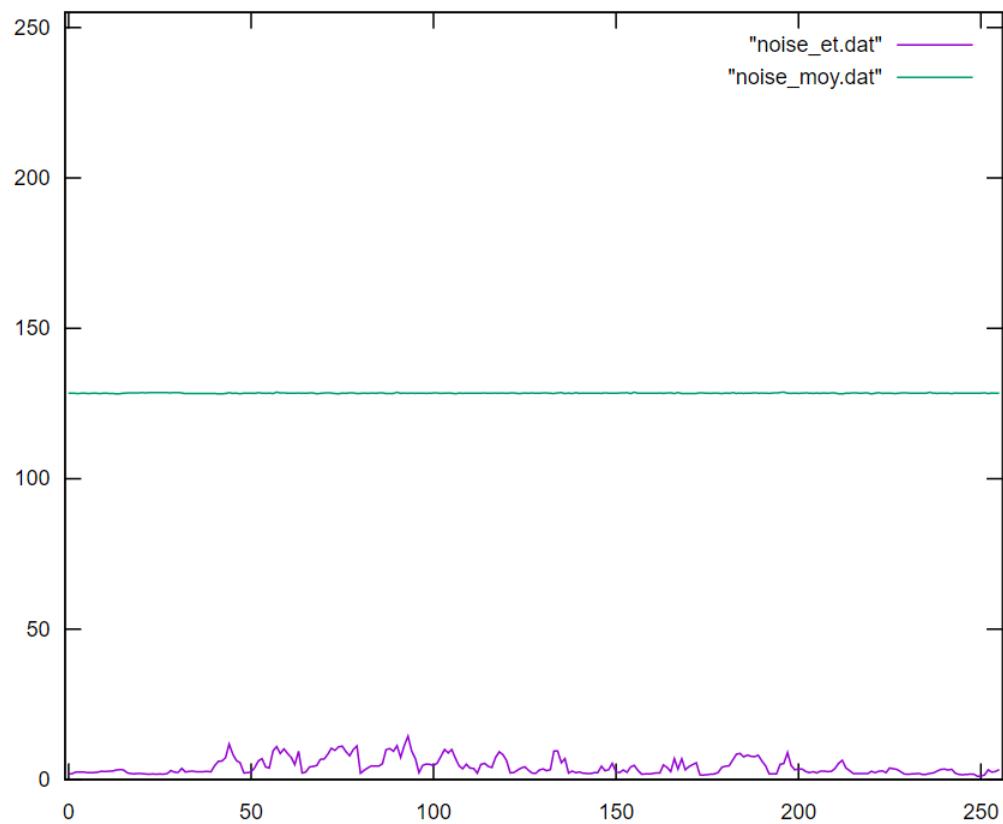
Que l'on va ensuite flouter :



Pour en extraire l'image de bruit suivante :



On peut ensuite partitionner cette image en 256 cadrans (16x16) pour en extraire les courbes suivantes sous gnuplot :



On peut ainsi dresser le même constat que pour l'image initiale, dont on a extrait le bruit, concernant les écarts types et la moyenne de chacun des cadans.

Les écarts types aux bords de l'image semblent varier plus faiblement (courbe plate), étant donné que le focus de l'image est fait sur le personnage en son centre.

## Patie 4 :

On va dans cette partie nous intéresser à une image retouchée afin de s'assurer que l'on arrive bien à détecter la retouche grâce au bruit.

Pour cela, nous allons prendre l'image 4\_Red\_tower.pgm :



À l'œil nu, on peut déjà remarquer que quelque chose cloche avec la tour de droite (notre droite), car l'arbre sensé se situer devant elle est en réalité derrière (branches découpées). Pour en avoir le cœur net, on va réitérer le même traitement qu'à la partie précédente, appliqué à cette image.

Premièrement, on va flouter l'image grâce à l'algorithme de blur.cpp donné.

Image floutée :



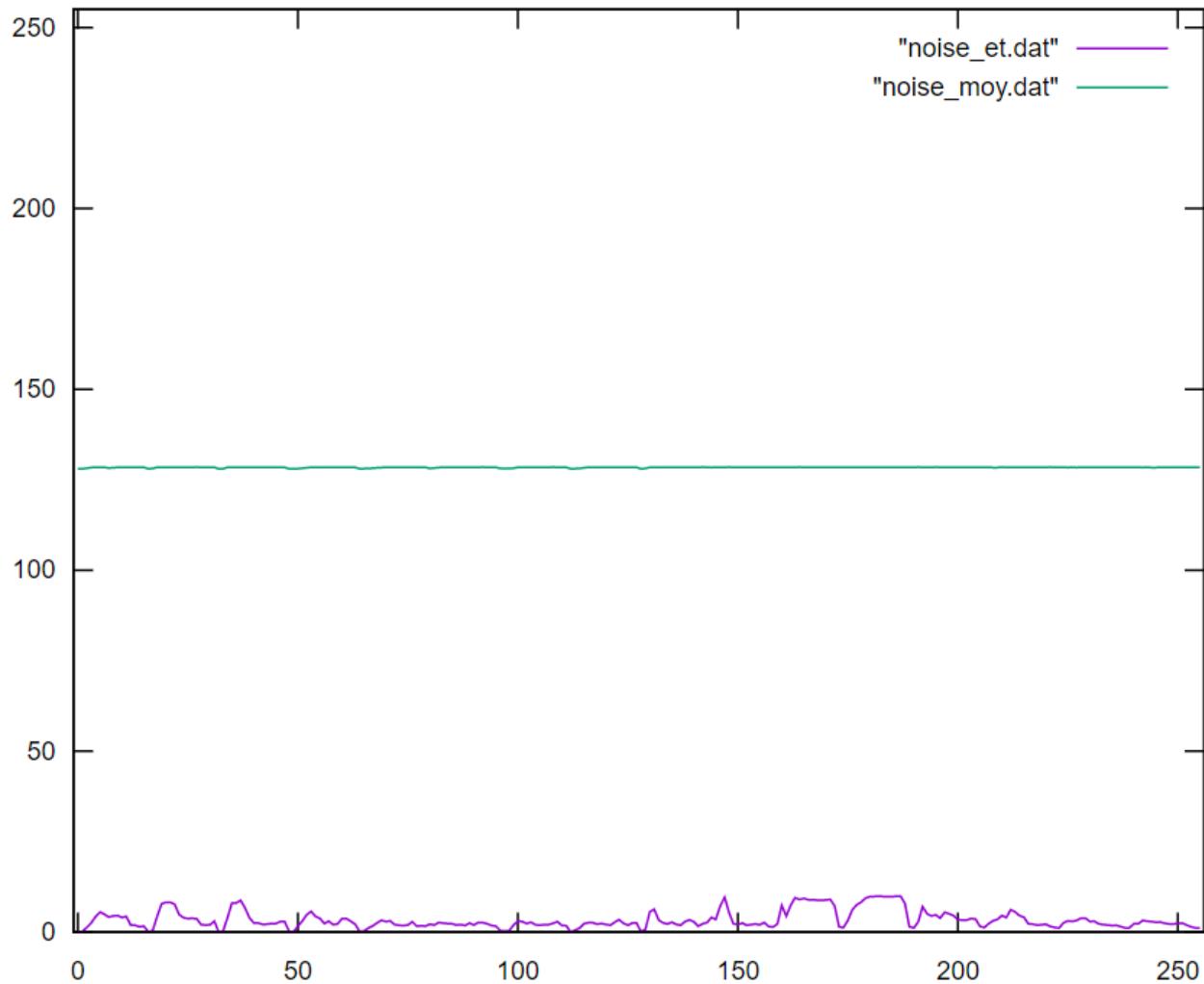
On peut ensuite déterminer le bruit de cette image comme différence entre l'image de base et l'image floutée.

Image du bruit obtenue :



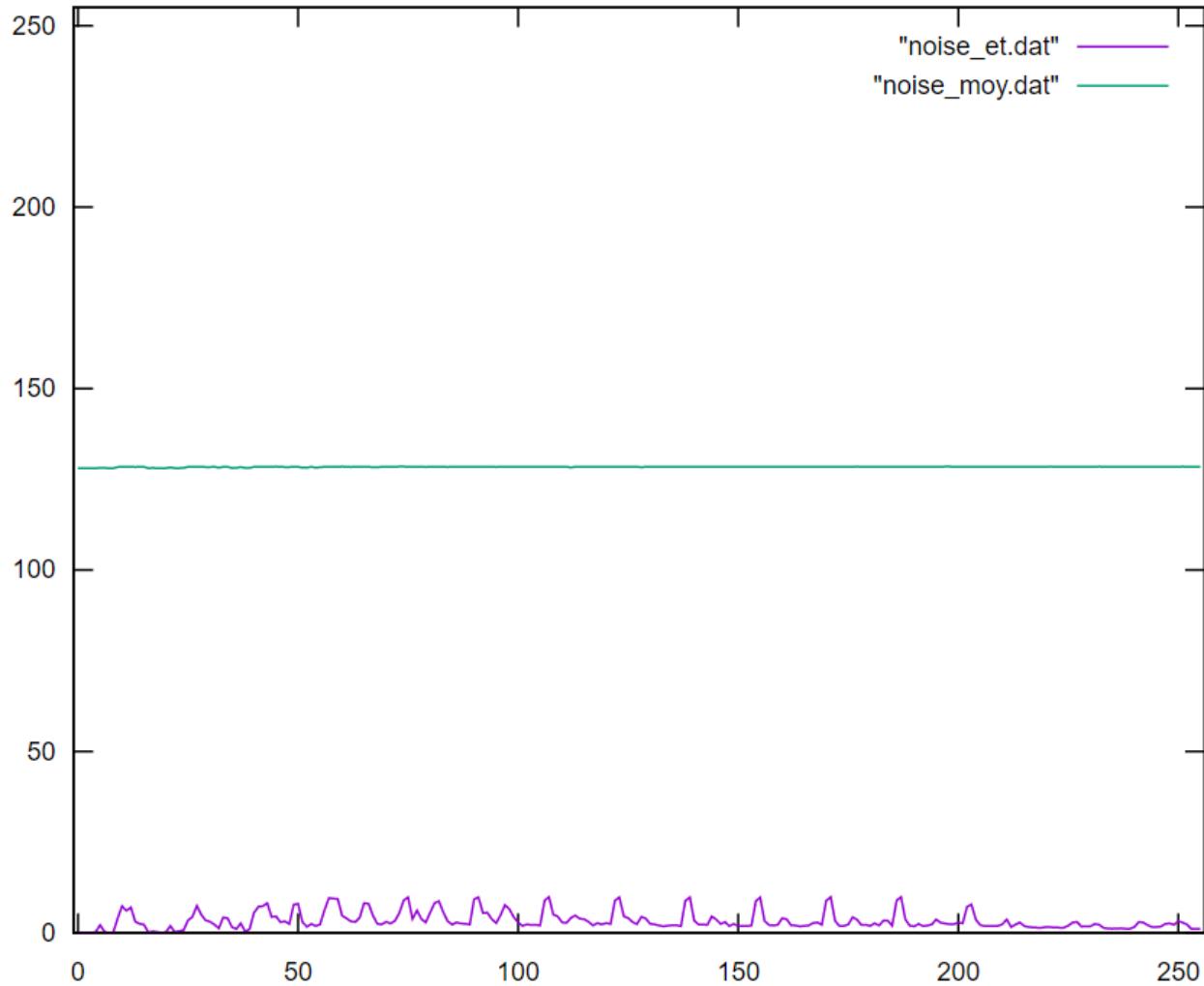
On se rend déjà compte que l'image du bruit est beaucoup moins expressive que pour les autres images analysées jusque-là.

On obtient, en partitionnant cette image en 256 cadrans, les deux courbes suivantes sous gnuplot :



Si la courbe de la moyenne semble toujours constante, on constate une variation au niveau de l'écart type. En effet, entre 0 et 50 et entre ~120 et 200, les variations semblent suivre un pattern différent (pics plus importants, présence de motifs répétitifs).

Entre 50 et 120 et entre 200 et 255, les écarts types semblent former une courbe constante en termes de variation. Cela n'est pas à première vue explicable autrement que par une retouche faite sur l'image initiale.



En analysant les 256 cadrans non plus de haut en bas puis de gauche à droite (en colonne), mais de gauche à droite puis de haut en bas (en ligne), on obtient la courbe ci-dessus. On se rend également compte que si la variation des écarts types semble suivre un motif régulier de 0 à 200, de 200 à 255, on obtient une variation quasi nulle d'un cadran à un autre (courbe constante). Il semble donc y avoir une différence majeure entre le haut de l'image et la bas, d'après l'analyse du bruit.

Pour que ces résultats soient plus parlants et afin de clairement identifier les zones clefs qui semblent être modifiées, on peut dresser une image correspondant à la valeur de l'écart type normalisé entre 0 et 255 de chacun des cadrans (écart type plus parlant que moyenne toujours constante).

En utilisant seulement 256 cadrans, on obtient l'image suivante :

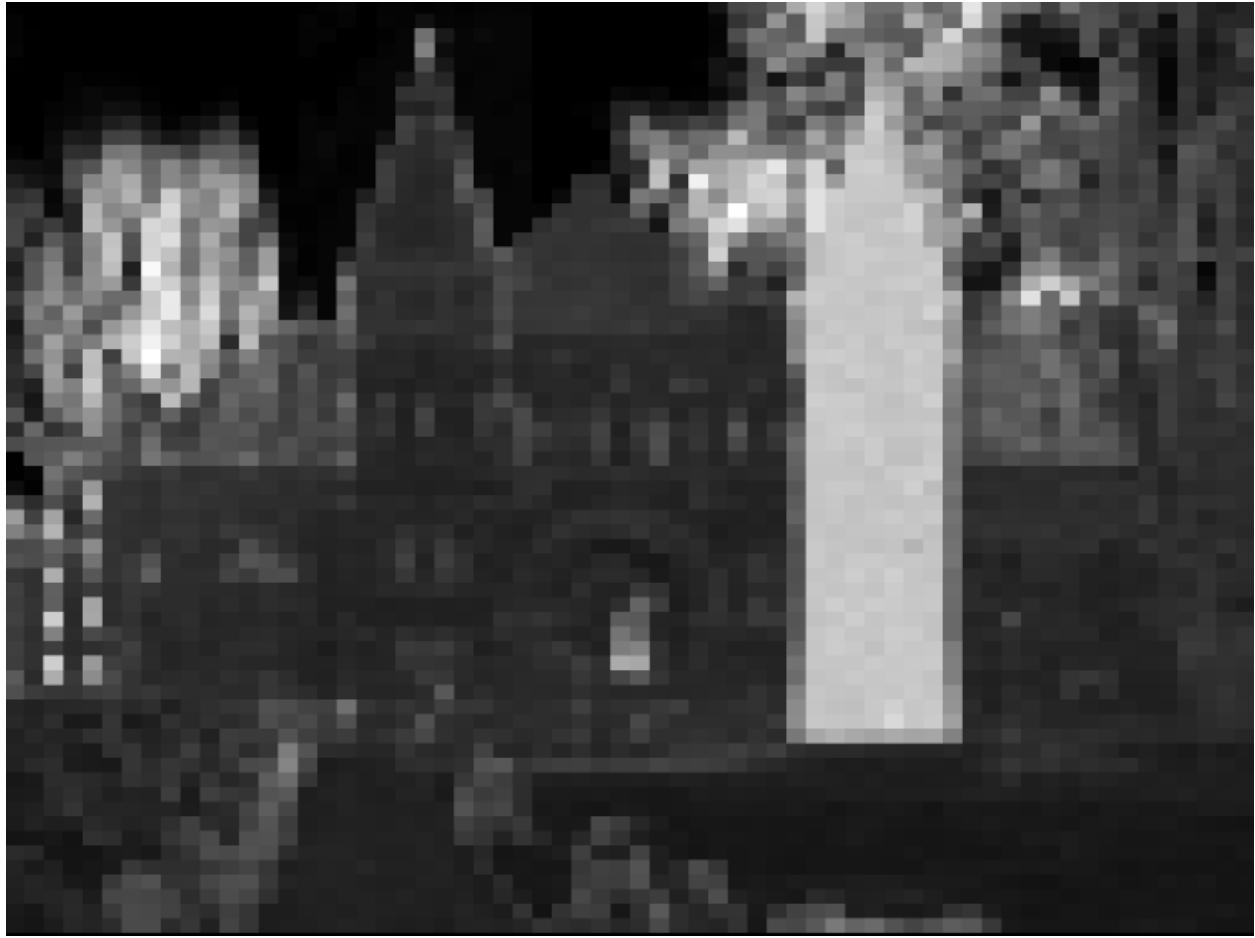


On distingue déjà une variation d'écart types entre les zones claires (blanches) et les zones plus sombres (gris foncé).

On peut augmenter encore le nombre de cadrants afin d'augmenter la précision de détection de la zone falsifiée (32x32 pour des cadrants de taille 102x76) :

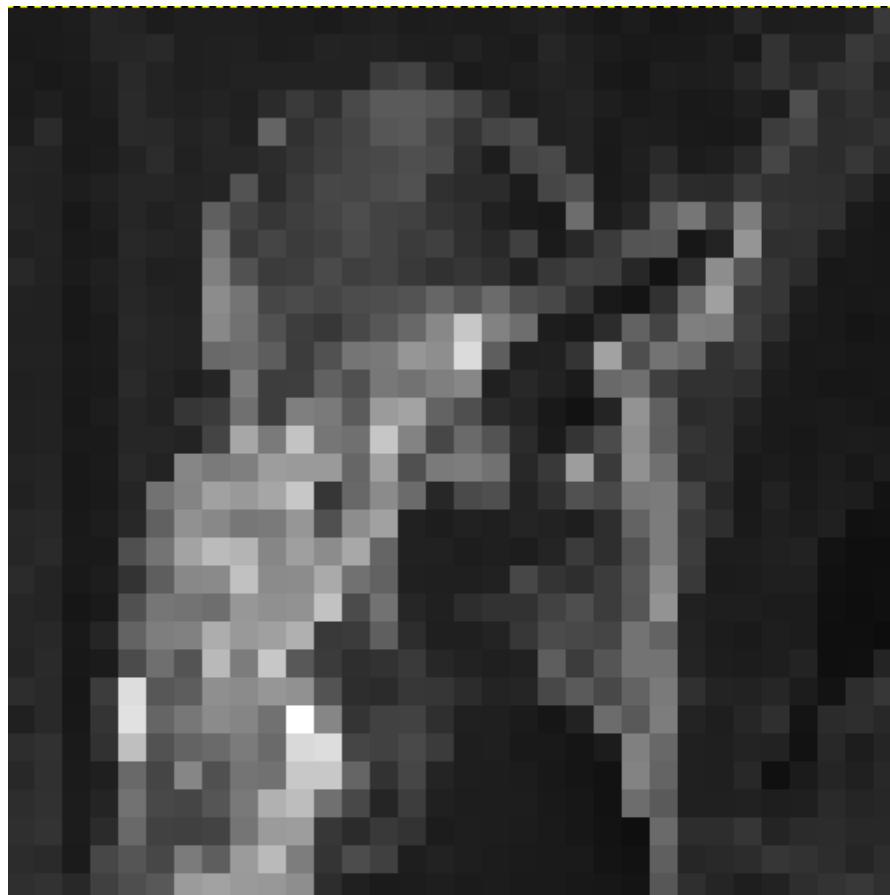


Enfin, en prenant 64x64 cadrans pour des tailles de cadrans de 51x38, on obtient l'image suivante (il est inutile de trop chercher à augmenter le nombre de cadrans pour raffiner l'image obtenue, car il faut un certains nombre de valeurs dans chacun des cadrans pour opérer un calcul de moyenne et d'écart types significatif) :



Avec cette image, on se rend bien compte que la seconde tour de l'image semble avoir été retouchée, car elle possède un écart type (image de bruit) qui est visuellement différent du reste de la bâtie. Il va de même pour l'arbre situé en haut à gauche de l'image, qui peut se traduire par l'écart entre la couleur de l'arbre (noir) et celle du ciel en fond (gris clair).

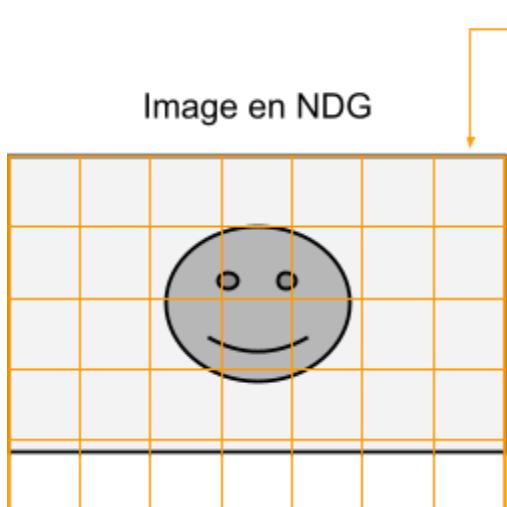
On peut répéter exactement le même processus mais pour une image à priori non falsifiée, comme l'image de Lena dont on en extrait le bruit (32x32 cadrans de taille 16x16) :



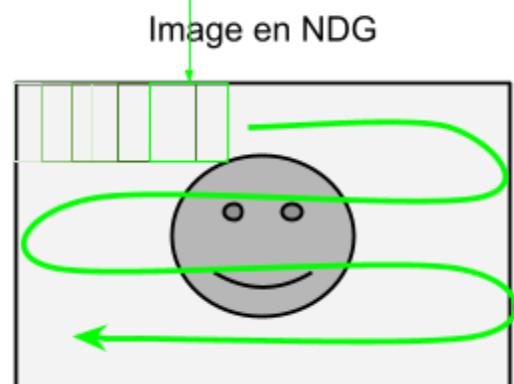
On se rend compte que la plupart des variations d'écart types correspondent en réalité à des variations de couleurs mesurables sur l'image d'origine (différence de luminosité), et non à des variations liées à une falsification.

## Patie 5 :

Pour plus de précisions, on peut changer notre approche de partitionnement dyadique par cadrans de l'image de bruit pour une fenêtre glissante.



CADRAN



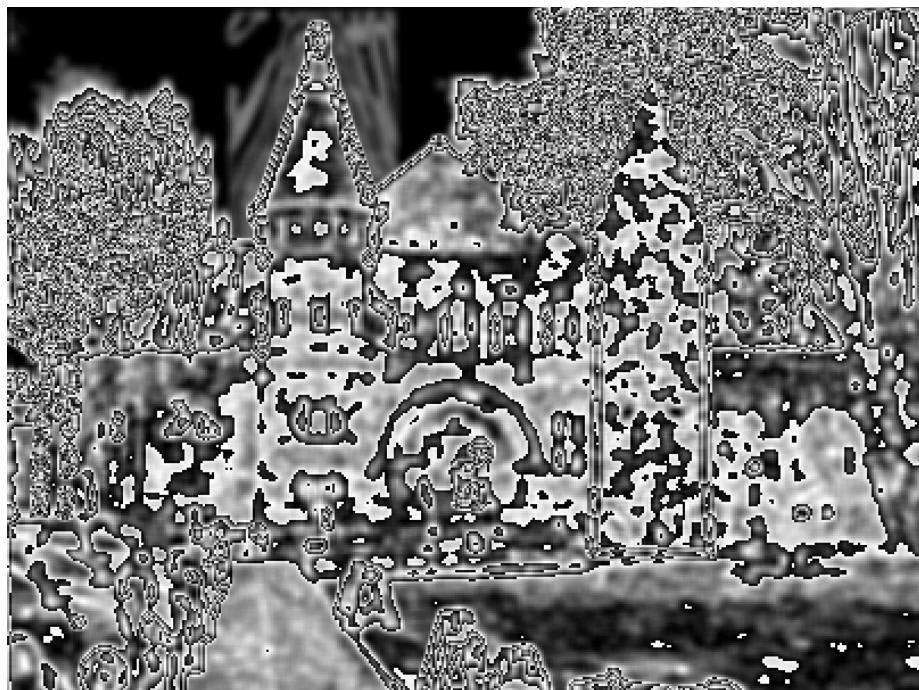
Partitionnement dyadique

Fenêtre glissante

On peut par exemple prendre une fenêtre de taille fixe et de pas de déplacement  $\text{largeurFenetre}/4$  pour x et  $\text{hauteurFenetre}/4$  pour y.

Si on reprend l'image de bruit de l'image probablement falsifiée du bâtiment avec les deux tours et qu'on utilise un algorithme implémentant la stratégie précédemment détaillée, on se retrouve avec les images suivantes :

-Taille de fenêtre 32x32



On se rend compte que le bruit autour de la tour la plus à gauche de l'image semble différent du reste de l'image, comme si elle semble avoir été retouchée.

Les arbres partagent visuellement le même bruit, de même pour le reste de la bâtisse, à l'exception de la tour à droite qui semble plus ébruitée.

-Taille de fenêtre 64x64



Les soupçons concernant la tour à gauche se confirment, on voit clairement une aura qui entoure le haut de celle-ci, qu'il n'y a pas pour les autres éléments de l'image.

L'autre tour possède un bruit bien plus clair que le reste du bâtiment, qui confirme ce que l'on a pu voir jusque là.

Enfin, on se rend compte que plus on augmente les dimensions de la fenêtre, et plus on perd en précision de rendu graphique et donc en discernement de zones falsifiées.

-Taille de fenêtre 128x64



Le principal changement par rapport à une grille de cadrans (partitionnement dyadique) est qu'un pixel peut obtenir des valeurs de plusieurs cadrans, et non d'un seul.

Toutes ces images sont relativement claires au niveau de leur rendu. En effet, j'ai calculé pour chacun des cadrans leur écart type, que j'ai ensuite appliqué, via une normalisation (/et max \* 255), à l'ensemble des éléments du cadran par addition successive (sans faire de moyenne ensuite).

Si je fais une moyenne des valeurs normalisées de chacun des cadrans incidents à un pixel, j'obtiens des résultats beaucoup plus sombres (fenêtre de taille 64x64) :

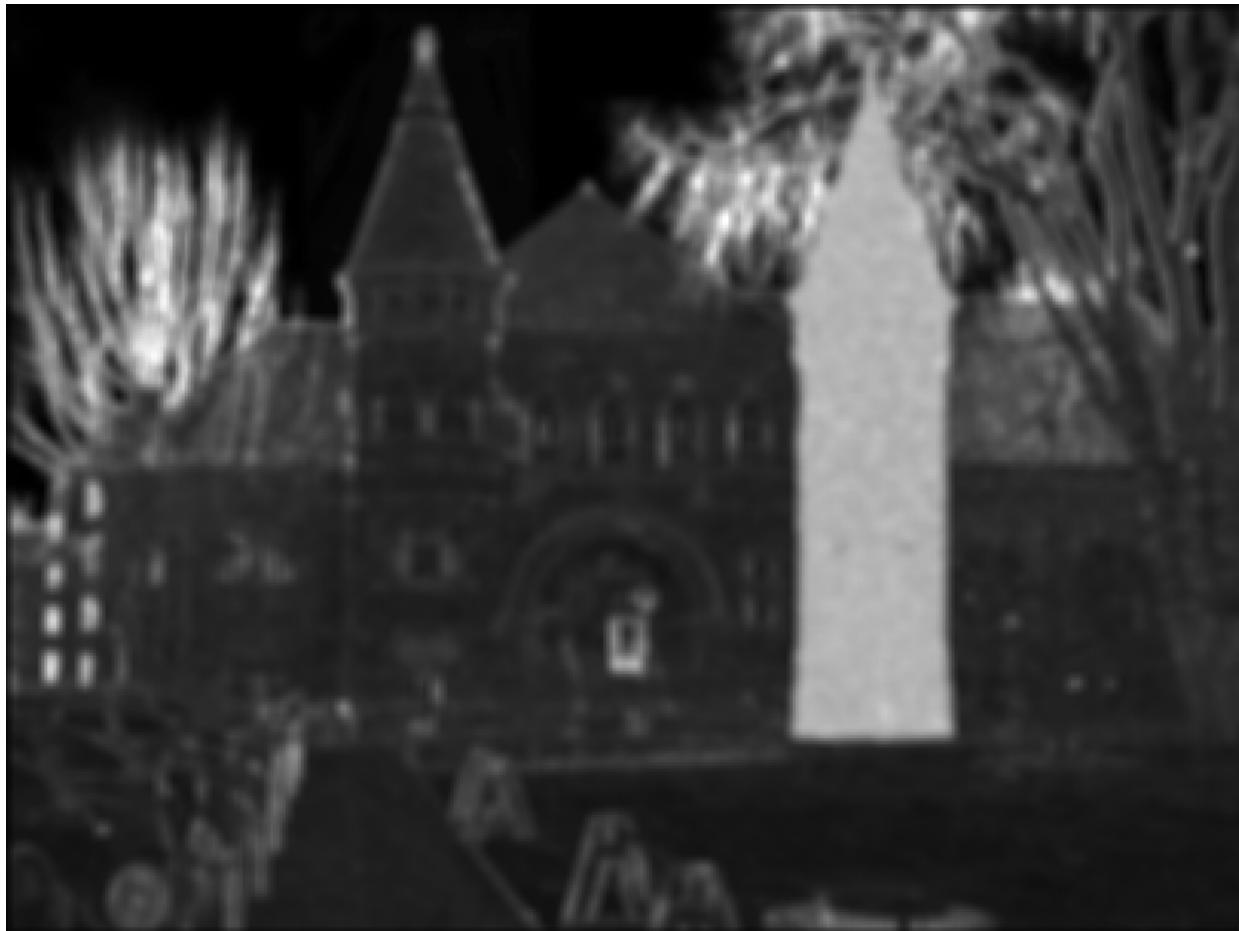


On ne peut guère extraire d'informations de cette image.

On pourrait aussi ajouter chacun des écart types des cadrans dans un tableau intermédiaire modélisant les pixels, puis calculer l'écart type maximal (maximum des écart types des cadrans additionnés entre eux).

On peut ensuite renseigner la valeur des pixels de l'image de sortie comme la valeur normalisée par l'écart type maximal \*255.

Pour des tailles de fenêtres de 32x32, on a :



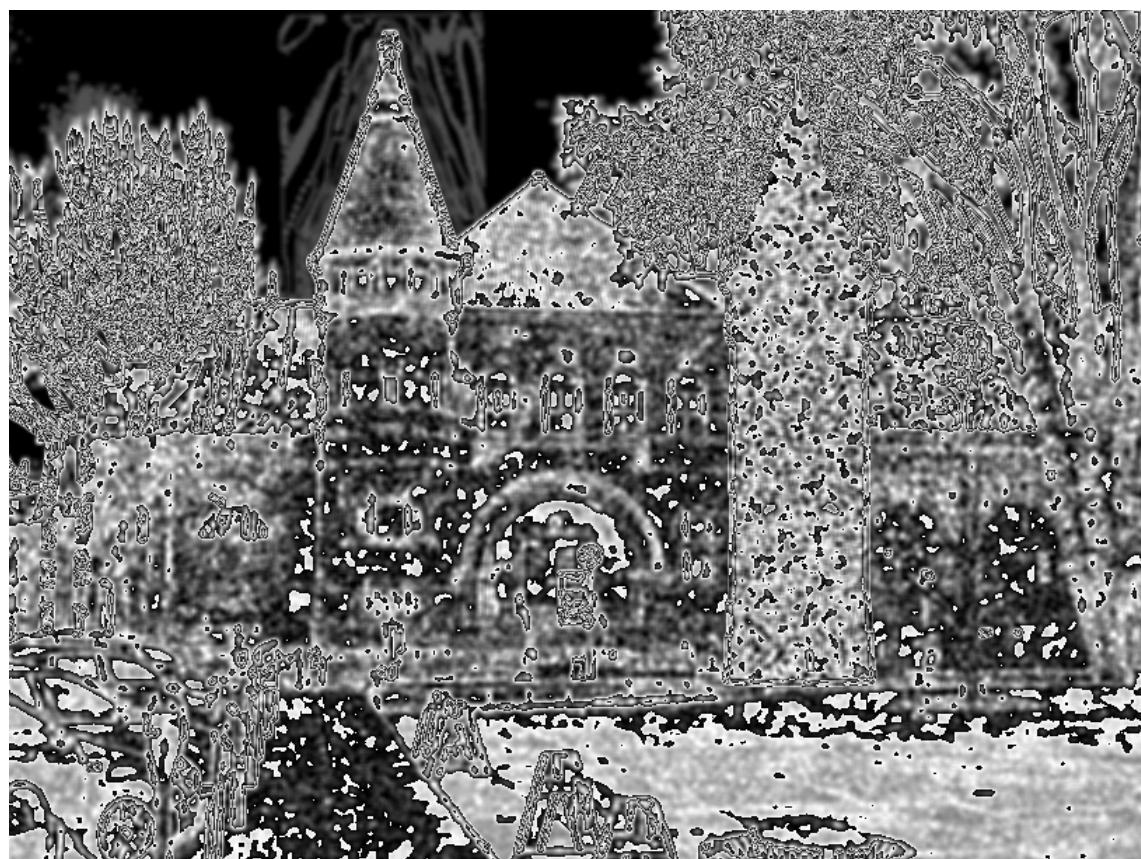
Cette image nous apporte plus de précisions pour les mêmes dimensions de cadrans (32x32) avec l'ancienne approche de partitionnement dyadique.

On remarque bien que la tour à droite ne possède clairement pas le même bruit que le reste du bâtiment.

On se rend compte par ailleurs que plus on augmente la taille de la fenêtre, et plus le résultat de l'image sera flou. En effet, pour des fenêtres de taille 64x64, on obtient :



Si on prend une fenêtre de résolution 16x16, on peut même apercevoir un début d'aura sur la tour à gauche, que l'on avait précédemment aperçu avec l'autre manière de concevoir le calcul pour la fenêtre glissante.



## Patie 6 :

Tout le traitement de détection de falsification d'image peut s'appliquer à des images colorées. Dans ce cas, il ne faut pas seulement composer avec une seule valeur de couleur, mais trois (rouge, vert et bleu). Ainsi, si on reprend l'image précédente mais colorée :



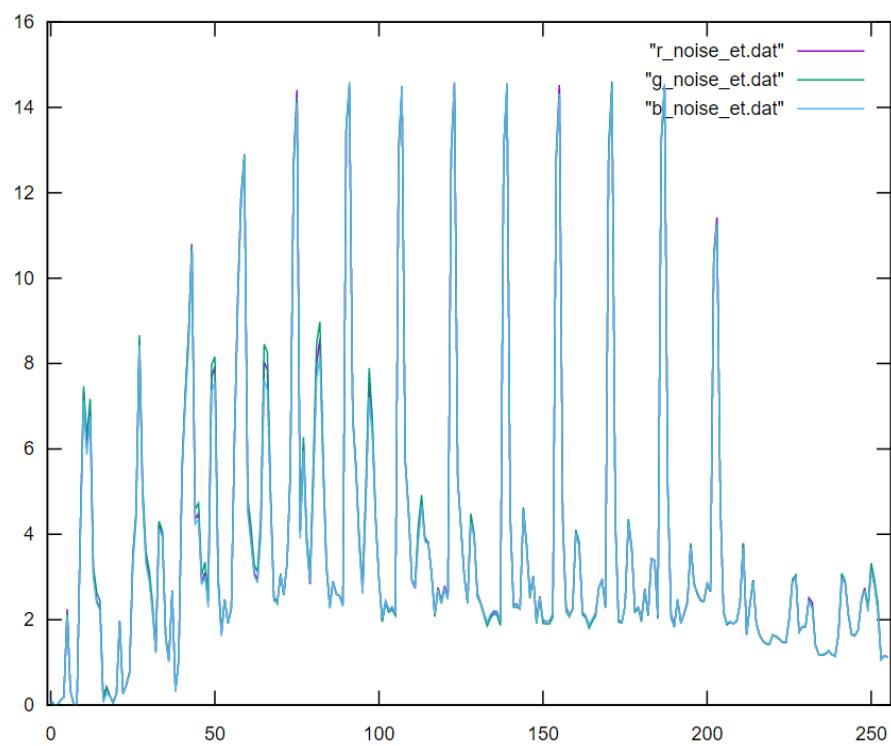
Que l'on peut flouter :



Pour en extraire l'image de bruit suivante :

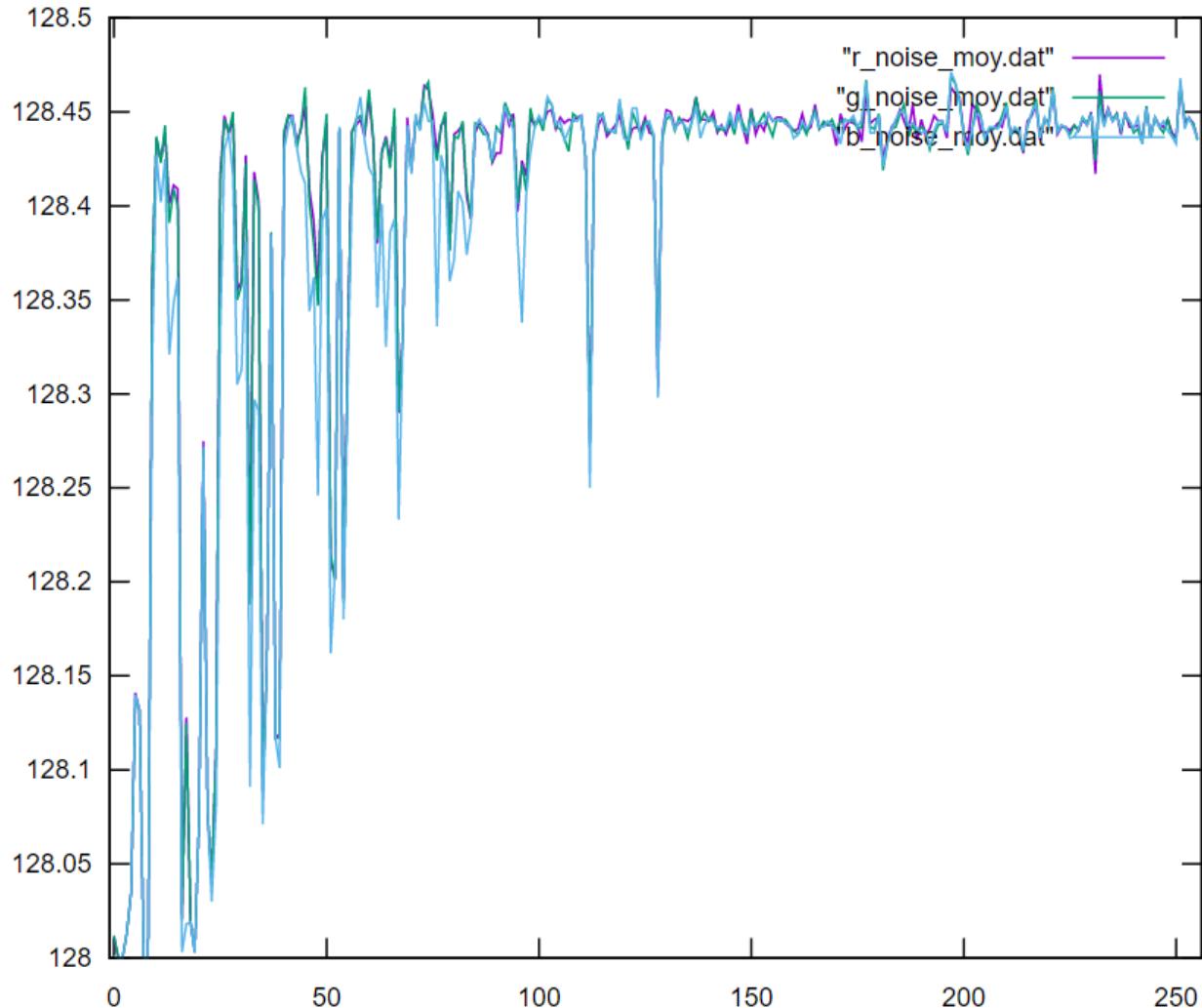


Cette image de bruit est similaire à l'image de bruit de l'image précédente en NDG.  
On peut tracer ensuite sur gnuplot la courbe de distribution des écart types de cette image de bruit pour 256 cadrans :



On se rend compte que les écarts types des différentes composantes de l'image de bruit ont des courbes qui suivent un pattern de répétition de pics et de creux similaires, sauf entre 200 et 255 (pics moins prononcés), qui pourrait traduire une falsification.

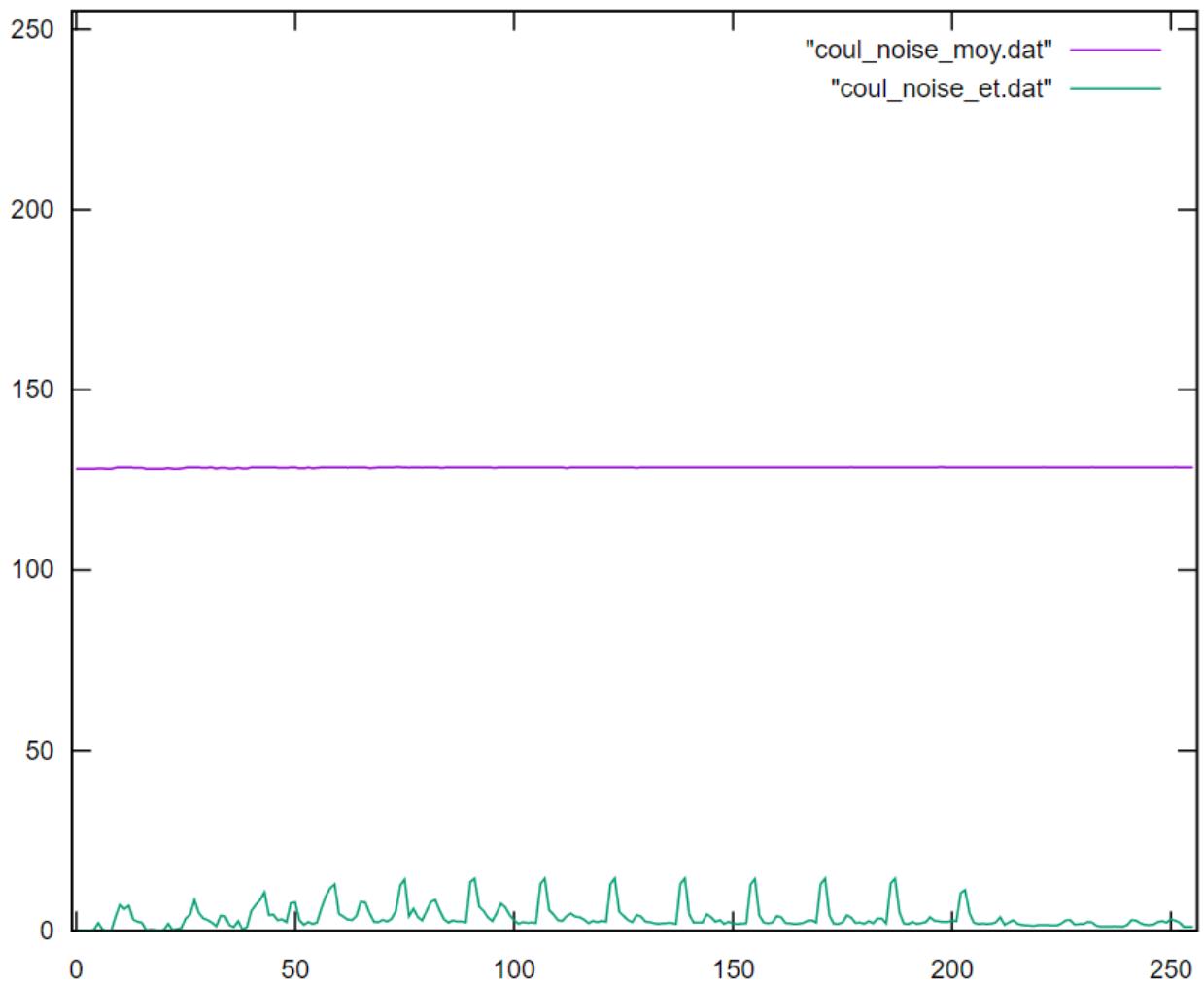
On peut aussi tracer la courbe de distribution des moyennes de l'image de bruit pour 256 cadrants :



On se rend compte, que comme pour l'image en NDG, la moyenne reste relativement constante (entre 128 et 128.5 ici).

Globalement, les différentes composantes ont des valeurs similaires pour le même cadran, que cela soit pour les écarts types ou pour les moyennes.

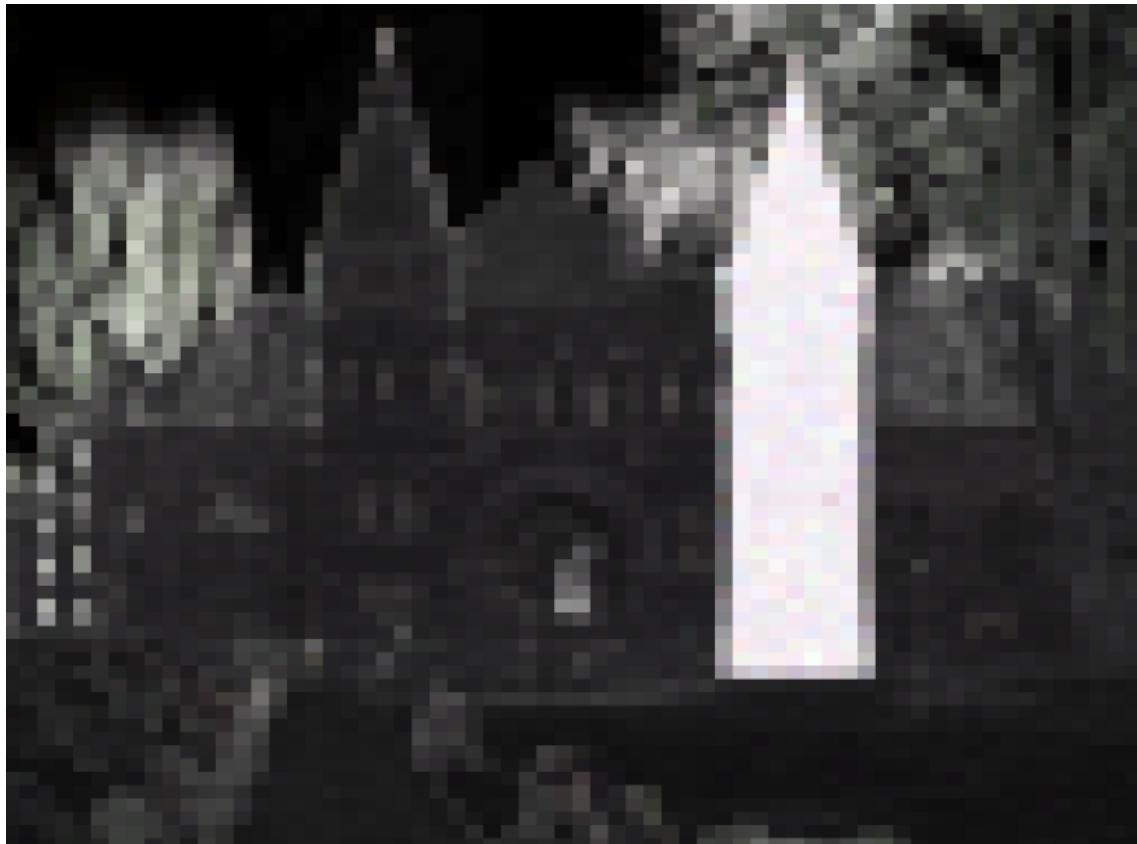
On peut enfin mélanger la valeur des trois composantes sous une seule courbes normalisée :



On retrouve en effet des courbes similaires aux courbes de l'image de bruit en NDG. Cela est dû en partie au fait que les deux images de bruits sont similaires (pour couleur comme pour NDG).

D'après cette analyse du bruit on peut en déduire, comme pour l'image en NDG, qu'il a effectivement une possibilité de falsification.

On peut par la suite utiliser l'algorithme de découpage dyadique de l'image de bruit pour produire une image des variances (pour des cadrants de taille 64x64) :



Ensuite, en utilisant l'algorithme par fenêtre glissante de taille 16x16 :



On remarque plus ou moins les mêmes choses qu'avec la même image en NDG.  
Enfin, on pourrait également augmenter le bruit initial appliqué à l'image :



Qui nous permet d'obtenir l'image de bruit suivante :

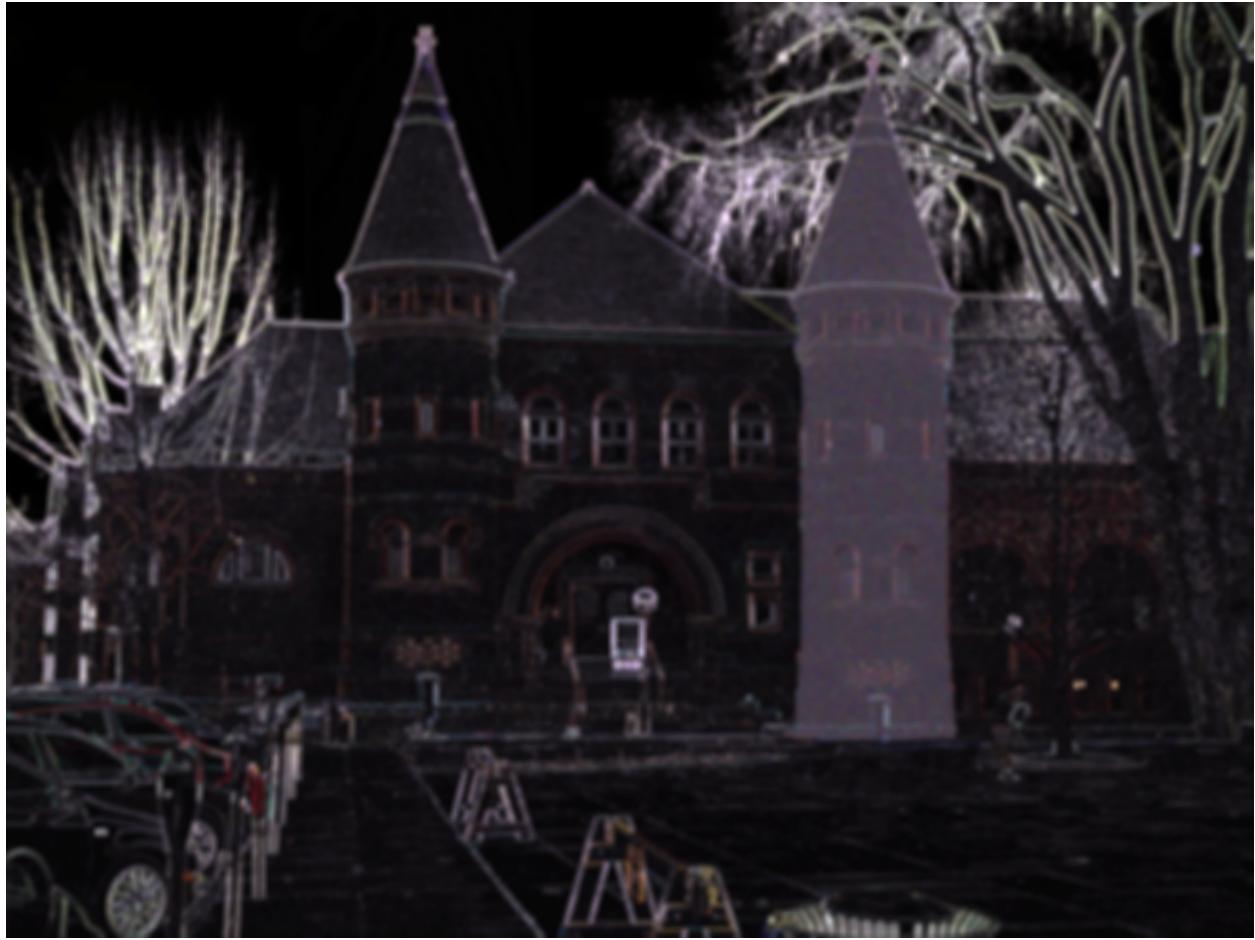


On peut ensuite l'utiliser afin de détecter les éventuelles falsifications. Dans un premier temps via l'algorithme de découpage dyadique de l'image de bruit, pour produire une image des variances (pour des cadrans de taille 64x64) :



On remarque que la tour à droite semble toujours ressortir de l'image de bruit. L'image est cependant plus coloré.

On peut aussi passer via l'algorithme de fenêtre glissante de taille 16x16. On remarque alors la même chose que pour les exemples précédents, mais toujours avec plus de couleurs :



Ainsi, augmenter le bruit de départ ne permet pas dans ce cas là de repérer plus simplement les zones falsifiées d'une image, bien au contraire.

## Commandes Gnuplot utiles

```
//Définir une sortie sur une image SVG
set term svg
//Définir un fichier de sortie
set output 'NomFichier.svg'
//Définir une borne sur l'axe Y[/X]
set y[ /x]range [0:255]
//Dessiner dans la sortie définie un fichier .dat
plot "noise.dat" with lines
//Définir une fonction de Gaussienne
gauss(x,mu,sigma)=1/(sigma*sqrt(2.*pi))*exp(-(x-mu)**2./(2.*sigma**2.))
//Définir une fonction de Laplacienne
laplacienne(x,mu,sigma)=1./(2.*sigma/sqrt(2.)) * exp(-abs(x-mu)/(sigma/sqrt(2.)))
//Définir une fonction de maximum
max(x,y)= (x<y)? y : x
```