**Bruitage et débruitage d’image**

**I. Bruitage de l’image :**

1. Comprendre le bruit « Pepper and Salt ».

Le bruit « poivre-et-sel » : consiste à mettre, aléatoirement, plusieurs pixels aux valeurs 255

ou 0 (valeurs extrêmes de l’intervalle des niveaux de gris).

2. Partie A :

code:(pour le blanc)

#include <iostream>

#include <opencv2\core\core.hpp>

#include <opencv2\highgui\highgui.hpp>

using namespace std;

using namespace cv;

void salt(cv::Mat &image, int n) {

int i,j;

for (int k=0; k<n; k++) {

// rand() is the MFC random number generator

i= rand()%image.cols;

j= rand()%image.rows;

if (image.channels() == 1) { // gray-level image

image.at<uchar>(j,i)= 255;

} else if (image.channels() == 3) { // color image

image.at<cv::Vec3b>(j,i)[0]= 255;

image.at<cv::Vec3b>(j,i)[1]= 255;

image.at<cv::Vec3b>(j,i)[2]= 255;

}

}

}

int main()

{

srand(cv::getTickCount()); // init random number generator

//cv::Mat image= cv::imread("Monalisa.jpg",0);

/////////:

std::string path = "C:\\monalisa.jpg";

// Ouverture de l'image

cv::Mat image = cv::imread (path);

salt(image,3000);

//cv::namedWindow("");

//cv::imshow("Image",image);

cv::namedWindow ("tp1!", CV\_WINDOW\_AUTOSIZE);

cv::imshow ("Hello World!",image);

cv::imwrite("salted.bmp",image);

cv::waitKey(15000);

return 0;

}

Résultat:

code:(pour le noir):

#include <iostream>

#include <opencv2\core\core.hpp>

#include <opencv2\highgui\highgui.hpp>

using namespace std;

using namespace cv;

void salt(cv::Mat &image, int n) {

int i,j;

for (int k=0; k<n; k++) {

// rand() is the MFC random number generator

i= rand()%image.cols;

j= rand()%image.rows;

if (image.channels() == 1) { // gray-level image

image.at<uchar>(j,i)= 255;

} else if (image.channels() == 3) { // color image

image.at<cv::Vec3b>(j,i)[0]= 0;

image.at<cv::Vec3b>(j,i)[1]= 0;

image.at<cv::Vec3b>(j,i)[2]= 0;

}

}

}

int main()

{

srand(cv::getTickCount()); // init random number generator

//cv::Mat image= cv::imread("Monalisa.jpg",0);

/////////:

std::string path = "C:\\monalisa.jpg";

// Ouverture de l'image

cv::Mat image = cv::imread (path);

salt(image,3000);

//cv::namedWindow("");

//cv::imshow("Image",image);

cv::namedWindow ("tp1!", CV\_WINDOW\_AUTOSIZE);

cv::imshow ("Hello World!",image);

cv::imwrite("salted.bmp",image);

cv::waitKey(15000);

return 0;

}

Résultat:

resultat:(pour le n=1000):

interprétation:

Plus le n augmente plus le bruitage de l'image est important.

**II. Filtrage d’image: #include <iostream>**

#include <opencv2\core\core.hpp>

#include <opencv2\highgui\highgui.hpp>

#include "opencv2/imgproc/imgproc\_c.h"

#include "opencv2/imgproc/imgproc.hpp"

#include <cv.h>

#include <highgui.h>

#define h 280

#define w 320

int main()

{

//Loading the image

IplImage\* img=cvLoadImage("C:\\lena.jpg");

//Image Variables

IplImage\* image=cvCreateImage(cvSize(w,h),8,3);

IplImage\* blurimg=cvCreateImage(cvSize(w,h),8,3);

IplImage\* medimg=cvCreateImage(cvSize(w,h),8,3);

IplImage\* gaussimg=cvCreateImage(cvSize(w,h),8,3);

//Resize the image

cvResize(img,image,CV\_INTER\_LINEAR);

//Windows

cvNamedWindow("Orginal Image",CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("Blur",CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("Median",CV\_WINDOW\_AUTOSIZE);

cvNamedWindow("Gaussian",CV\_WINDOW\_AUTOSIZE);

//Different Smoothing

//Blurring the image

cvSmooth(image,blurimg,CV\_BLUR,7,7);

//Median smoothing

cvSmooth(image,medimg,CV\_MEDIAN,7,7);

//Gaussian Smoothing

cvSmooth(image,gaussimg,CV\_GAUSSIAN,7,7);

//Showing the Images

cvShowImage("Orginal Image",image);

cvShowImage("Blur",blurimg);

cvShowImage("Median",medimg);

cvShowImage("Gaussian",gaussimg);

//Escape Sequence

cvWaitKey(0);

//CleanUp

cvReleaseImage(&img);

cvDestroyAllWindows();

}

blur

median

gaussian

Quel est le filtre qui améliore le plus la qualité de votre image ? Justifier votre réponse.

le mielleur filtre est le gaussian car sur une image bien

contracté il fait un effet de flou sans faire perdre les

détails de l'image originale. la justification de la réponse est à partir de la constations entre la différence entre les images filtrées.