Université EuroMed de Fès

Ecole de l'ingénierie digitale et de l'intelligence artificielle

TP De HSLA IMAGES

Espace colorimétrique de l'image HSLA

Eddouks Oumayma & El Omary Imane

objectifs :

1. comprendre plus sur l'héritage dans la programmation orientée objet et diverses formes d'héritage

2. Comprendre l'espace colorimétrique HSL

Sommaire :

La Classe PNG

Dans image_manip.zip, on donne un projet qui contient une classe appelée PNG qui implémente la manipulation d'images de base comme :

- **Lecture d'** une image du système.
- **Ecriture d'** une image dans le système.
- **Accès aux pixels** de cette image.

IMAGE :

On a créé une classe nommée Image qui hérite de la PNG classe. De plus, la classe IMAGE héritera de tous les attributs et membres de la PNG classe.

On a ajouté les méthodes suivantes :

- Image (string filename) : un constructeur spécial qui charge l'image à partir du fichier filename
- Lighten (double amount) : modifie la luminance de chaque pixel par quantité.

exemple : **Saturate** : modifie la luminance par quantité

RotateColor(double angle) : ajoute la valeur de l'angle à chaque pixel.

stanford_image_manip/image...* | Image::saturate(double) -> void

```cpp
#include "image.h"
# include "PNG.h"
Image::Image(string filename):PNG(){
    readFromFile(filename);}
void Image::lighten(double amount)
{ for(unsigned i=0;i< width();i++){
        for(unsigned j=0;j< height();j++) {
          HSLAPixel & P= getPixel(i,j);
          P.l +=amount;
            //ne pas depasser 1
            P.l = (P.l <1)? P.l :1;
            //ne pas depaser 0
            P.l=(P.l<0) ? 0 : P.l;}
}
}
void Image::saturate(double amount){
    for(unsigned i=0;i< width();i++)
        for(unsigned j=0;j< height();j++)
        {
            HSLAPixel & P= getPixel(i,j);
              P.s +=amount;
            //ne pas depasser 1
            P.s = (P.s <1)? P.s :1;
            //ne pas depaser 0
            P.s=(P.s<0) ? 0 : P.s; }
}
void Image::rotatecolor(double angle){
    for(unsigned i=0;i< width();i++)
        for(unsigned j=0;j< height();j++){
            HSLAPixel & P= getPixel(i,j);
            P.h += angle;
            //S'assurer  que P.h < 360
            while( P.h > 360)
                P.h -= 360;
            //S'assurer  que P.h >0
            while( P.h <0)
                P.h += 360; }
}

```

image.h | <Select Symbol>

```cpp
#ifndef IMAGE_H
#define IMAGE_H
#include"PNG.h"


class Image:public PNG
{
public:
    using PNG::PNG;
    Image(string);
    //LIGHTEN
    void  lighten(double amount=.1);
    void saturate(double amount=.1);
    void rotatecolor (double angle);
};

#endif // IMAGE_H

```

Résultat de lighten:

Résultat de saturate:

Résultats de RotateColor:

NIVEAU GRIS (GrayScale) :

On a écrit **une classe simple GrayScale** qui hérite **de la Image Classe**. Il s'agit d'une classe simple qui élimine toutes les couleurs et représente l'image en utilisant uniquement un GrayScale niveau.

grayscale.h

```
#ifndef GRAYSCALE_H
#define GRAYSCALE_H
#include"PNG.h"

class Grayscale:public PNG
{
public:
    using PNG::PNG;
    Grayscale(string );

};

#endif // GRAYSCALE_H
```

grayscale.cpp <Select Symbol>

```cpp
#include "grayscale.h"

Grayscale::Grayscale(string filename):PNG(){
    readFromFile(filename);

      for (unsigned i = 0; i <width(); i++) {
        for (unsigned j = 0; j < height(); j++) {
          HSLAPixel &P = getPixel(i, j);

          P.s = 0;
        }
      }


    }

```

Résultat :

Illini :

On a créé une classe appelée Illini qui hérite de la Image Classe .

La classe a un constructeur qui accepte ces deux couleurs

- Illini(string filename, int color1, int color2) ;

Un Illini image n'a que deux couleurs définies en tant d'attributs

- Color1= par défaut est égal à illini orange =11
- Color2= par défaut est égal à illini blue =216

illini.h <Select Symbol>

```
1   #ifndef ILLINI_H
2   #define ILLINI_H
3   #include "PNG.h"
4
5   class Illini: public PNG
6   {
7   public:
8        using PNG::PNG;
9       Illini(string,int,int);
10   private:
11      int color1;
12      int color2;
13
14  };
15
16  #endif // ILLINI_H
17
```

stanford_image_manip/illini.cpp | Illini::Illini(std::__cxx11::string, int, int) -> void

```cpp
#include "illini.h"
#include "math.h"

Illini::Illini(string filename, int color1=11, int color2=216 ):PNG( )
{
    this->color1=color1;
    this->color2=color2;
    readFromFile(filename);

    for (unsigned i = 0; i< width(); i++){
        for (unsigned j=0; j<height();j++){
          HSLAPixel& P = getPixel(i,j);
          if(P.h >=0 && P.h <=180){
              P.h=abs(P.h-color1)<= abs(P.h-color2)? 11 : 216;}
          else if(P.h <=360 && P.h >=180){
              P.h=abs(P.h-color2)<= (360-P.h+11)? 216 :11;

      }


    }
    }
}

```

Résultat de Illini :

Projecteur (spotlight) :

Une Spotlight image crée un **projecteur** centré sur un point donné **centerX , centerY** défini en tant qu'attributs.

Un projecteur ajuste la **luminance** d'un pixel en fonction de la distance euclidienne à laquelle le pixel est éloigné du centre en diminuant la luminance de **0,5 %** pour 1 pixel, jusqu'à **80 % de** diminution de la luminance.

Par exemple, un pixel éloigné du centre de 3 pixels au-dessus et de 4 pixels à droite aura un total de $\sqrt{3^2 + 4^2} = \sqrt{25}=5$ pixels de distance et sa luminance est diminuée de **2,5%** (0,975x sa valeur d'origine). A une distance de plus de **160** pixels, la luminance sera toujours diminuée de **80%** (0,2x sa valeur d'origine).

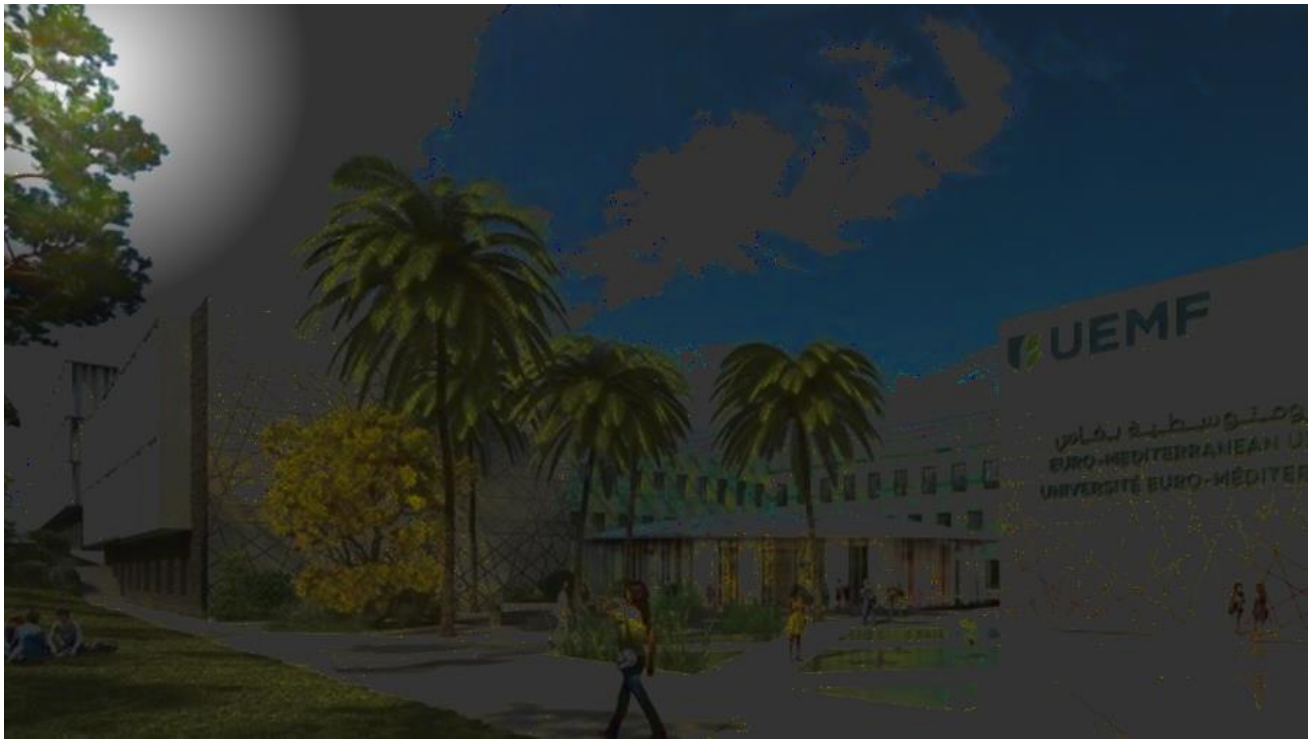
spotlight.h <Select Symbol>

```
#ifndef SPOTLIGHT_H
#define SPOTLIGHT_H
#include"PNG.h"
#include "HSLAPixel.h"

class Spotlight : public PNG
{
public:
    Spotlight(string filname );
    //int xCenter();
    //int yCenter();
    PNG createSpotlight(PNG img, int centerX, int centerY);

private:
    int xCenter;
    int yCenter;
};

#endif // SPOTLIGHT_H
```

stanford_image_manip/spotlig... | Spotlight::createSpotlight(PNG, int, int) -> PNG

```cpp
#include "spotlight.h"
#include "math.h"
#include "image.h"
#include"HSLAPixel.h"
#include"PNG.h"
Spotlight::Spotlight(string filname ):PNG()
{
    readFromFile(filname);
}
 PNG Spotlight :: createSpotlight(PNG image, int centerX, int centerY) {
     int xx=0, yy=0,z;
     for (unsigned x = 0; x < image.width(); x++) {
             for (unsigned y = 0; y < image.height(); y++) {
             HSLAPixel & pixel = image.getPixel(x, y);
             xx=x-centerX;
             yy=y-centerY;
             z=sqrt((xx*xx)+(yy*yy));
             if(z<160)
                     pixel.l=pixel.l-.005*z*pixel.l;
             else
                     pixel.l = pixel.l - 0.8*pixel.l;
             }
         }
       return image;
     }
```

Résultat de Spotlight :



Main classe :

stanford_image_manip/main....* | <Select Symbol>

```
//#include <iostream>
#include "console.h"
#include "testing/SimpleTest.h"
#include "image.h"
#include "grayscale.h"
#include "illini.h"
#include "gwindow.h"
#include "catch.h"
#include "spotlight.h"
using namespace std;
/*//
 * This sample main brings up testing menu.
 */
int main() {
    Image Im("res/euromed_image.png");
   // appliquer un traitement
    Im.lighten(0.5);
     Im.saturate(0.5);
    Im.rotatecolor(200);
//sauvegarder
  Im.writeToFile("res/new_image.png");

    Grayscale gr("res/euromed_image.png");
    gr.writeToFile("res/new_image.png");
   Illini ill("res/euromed_image.png",11,216);
     ill.writeToFile("res/new_image.png");
    Spotlight sp("res/euromed_image.png");
    sp.createSpotlight(sp,50,50).writeToFile("res/new_image.png");


    return 0;

}
```

Conclusion :

Ce projet bien que simple constitue une de nos premières expériences de gestion de projet,gestion du temps et d'équipe qui sont des choses fondamentales pour un futur ingénieur.