Abdoulaye Djibril Barry E14B533C
Vadim Borisov E160722Y
Groupe: 486M

UNIVERSITÉ DE NANTES

Rapport De Projet Création Numérique (Processing)

But du projet : Dans ce projet nous avons voulu appliquer la notion de filtre sur une image que nous avons télécharger sur internet. Le but étant de changer la couleur de l'image à chaque fois qu'on reste appuyé sur la souris, la saturation du filtre augmente proportionnellement à la durée de la pression sur la souris. Optionnellement on voulait que l'utilisateur puisse avoir le droit de choisir son propre filtre à travers le panneau des options.

Construction de l'image:

Initialisation:

Nous avons utilisé un carré pour le filtre, pour définir la taille du côté de ce carré nous avons utilisé la variable *int carre = 80*, ainsi que (*x_debut*, *x_fin*, *y_debut*, *y_fin*) pour définir les coordonnées du carré à partir du positionnement de la souris. Ensuite nous avons besoin de stocker la saturation instantanée du filtre, pour cela on utilise les variables *float saturation* et *float saturationMax*. On a aussi besoin de stocker les composantes r,g,b des pixels pour lesquels on change la couleur, donc on met float r,g,b.

Void setup() :

Dans cette fonction on définit la taille de la fenêtre qui correspond à la taille de l'image chargé. Nous avons téléchargé l'image(https://www.pexels.com/photo/close-up-photograph-of-slices-orange-citrus-fruits-943632/) et l'avons importé à travers notre code en Processing en utilisant la fonction loadImage(). Et on positionne correctement l'image pour qu'elle recouvre toute la fenêtre avec image(img,0,0).

Void draw() :

Dans cette fonction on a principalement deux actions : quand l'utilisateur appuie sur la souris (fonction mousePressed()) et quand l'utilisateur arrête d'appuyer sur la souris (fonction mouseReleased()). Pour calculer les dimensions d'un carré où on va appliquer le filtre, on utilise la fontion de Processing constrain(). Ensuite on travaille avec le tableau des pixels que l'on charge dans le programme avec la fonction loadPixels(). On utilise la formule : *pos = x + y*img.width* pour représenter linéairement le tableau des pixels présents dans le carré filtré. On a crée le filtre noir-blanc, grâce à la décomposition de chaque pixel en 3 composantes : rouge, vert et bleu. Et en remplaçant la couleur de chaque pixels avec la formule: *color((r+g+b)/saturation)*

Difficultés :

On a essayé de créer les options pour que l'utilisateur puisse choisir le filtre à partir de boutons sur le panel à droite de l'image. La difficulté était de limiter l'application d'un filtre sur l'image sans toucher le panneau des options des filtres. Ainsi on voulait remplacer la forme de carré par une ellipse.