Rapport Projet Traitement d'Image Android

Cyril CAULONQUE

Table des matières

1 Présentation

Cette application permet d'appliquer des filtres basiques à des images prédéfinies ou provenant de la caméra. Elle permet aussi de les enregistrer dans la galerie. Elle demande 2 permisions : écrire dans la mémoire externe et prendre des photos. La version de l'api requise est la 23.
Le code de l'application comporte 4 activités : *MainActivity, HistoActivity, KernelActivity, SelectColorActivity* et utilise 2 classes *Traitement* et *TouchImageView*, ainsi que 2 fichiers renderscript *colorize.rs* et *grey.rs*.
L'activité main appelle toutes les autres. Toutes les fonctions de traitement d'image sont situées dans *Traitement.java*. A l'ouverture l'application propose 5 images, situées dans le dossier ressources. Dans le code, elles sont stockées dans un ArrayList de Bitmaps, car on veut pouvoir ajouter des bitmaps à celles déjà disponibles. L'image affichée à l'écran est la bitmap *mutableBitmap*, l'image de base est stockée dans *image_bitmap* afin de pouvoir annuler les modifications à tout moment.

2 Les algorithmes

2.1 Les algorithmes basiques

Les algorithmes basiques proposés par l'application sont :

- griser
- griser + extension de l'histogramme
- coloriser (teinte au hasard dans le format hsv)
- noir & blanc (bicolore)
- filtrer (griser toutes les couleurs trop éloignées de la couleur sélectionné (Distance Euclidienne))

```
distance = (int) Math.sqrt(Math.pow((R0 - R), 2) + Math.pow((G0 - G),
    2) + Math.pow((B0 - B), 2));
if (distance > seuil) {
     greylvl = (R + G + B) / 3;
     pixels[k] = (255 & 0xff) << 24 | (greylvl & 0xff) << 16 | (greylvl
         & 0xff) << 8 | (greylvl & 0xff);
}
```

- extension de l'histogramme sur les 3 canaux RGB.
- égalisation. On crée une lookuptable avec les niveaux de gris et on l'applique ensuite aux 3 couleurs.

2.2 Les convolutions

Les algorithmes de convolutions sont :

- flou (taille variable) Les coefficients sont tous les mêmes.

— flou gaussien (taille variable) Les coefficients sont répartis de manière plus circulaire :

```
coeff = Math.exp(-((double) (x * x + y * y) / ((double) (2 * n * n))))
    ;
```

n étant la rayon du noyau. x et y la posiition du coefficient au sein du noyau.

— gradient. La racine de $gradientX^2 + gradientY^2$. Valeur ramenée entre 0 et 255.

— laplace. Assez semblable au gradient mais noyau différent :

```
noyaulaplace = {{0, 1, 0}, {1, -8, 1}, {0, 1, 0}};
```

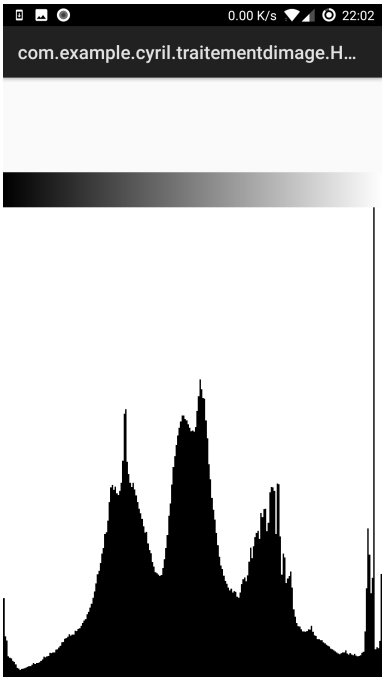
Les bords sont traités en reflétant l'image.

2.3 Renderscript

Il y a 2 scripts renderscript : 1 pour la fonction griser qui apporte une différence négligeable (133 639 micro secs réduit à 100 607 micro secs). La fonction la plus lente était la fonction colorize (chaque pixel est passé du format rgb au format hsv puis à nouveau rgb) avec un temps d'exécution de 10 340 087 micro secs. La version renderscript dure 339 104 micro secs, elle est donc 30 fois plus rapide.

3 Fonctionnalités Supplémentaires

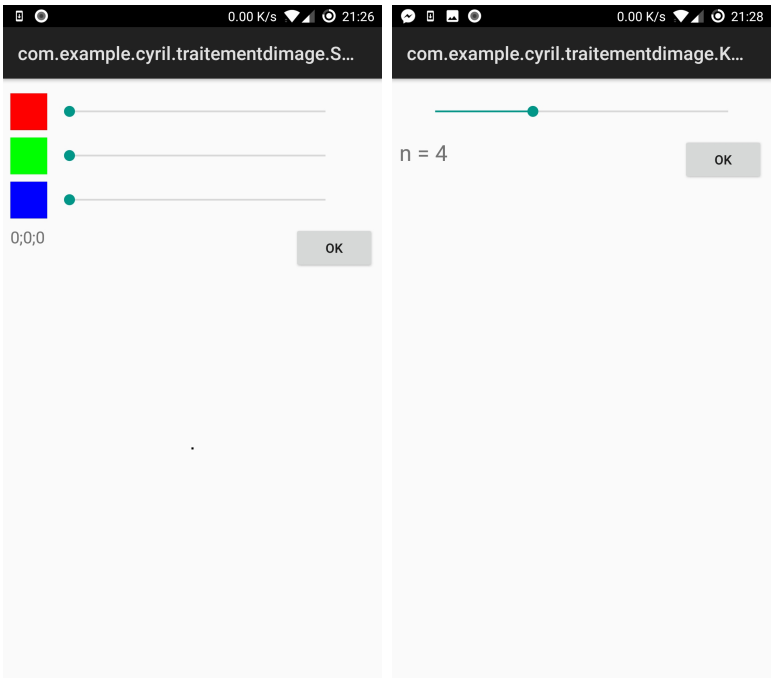
3.1 Visualiser l'histogramme



3.2 Prendre une photo

L'application permet de prendre une photo, celle ci est ajoutée à la fin de l'ArrayList des bitmaps, peut donc être modifiée comme une image classique.

3.3 Sélectionner une couleur (ou une taille)



Il est possible de sélectionner une couleur pour la fonction "filtrer" (qui était à la base choisie aléatoirement), et de choisir la taille du kernel pour les fonctions de flou.

3.4 Zoom au doigt

La fonctionnalité de zoom au doigt utilise à 100% une classe importée depuis un projet github. J'avais mes propres fonctions mais elles n'étaient pas totalement fonctionnelles, je n'arrivais pas à centrer le zoom sur le point entre les 2 doigts malgré de nombreuses tentatives et solutions différentes. J'ai donc décidé d'utiliser la classe TouchImageView.java trouvée sur internet.

3.5 Sauvegarder une image

L'application demandera la permission à l'utilisateur d'écrire dans la mémoire. Si la réponse est positive alors l'image en cours est enregistrée dans un sous dossier du dossier Pictures. Le fichier est nommé selon la date et l'heure.