Aymane ICHOU
M1 BIG DATA

Rapport de projet
Interface graphique et traitement d'images
Scilab

Sommaire

1. Introduction

L'objectif de ce projet est de concevoir une application graphique interactive sous Scilab permettant d'appliquer différents traitements sur des images de manière simple et visuelle.

L'utilisateur peut charger une image, la visualiser, l'exporter puis exécuter des opérations telles que la conversion en noir et blanc, le seuillage, l'amélioration du contraste, la rotation, ou encore l'affichage de l'histogramme.

Le projet repose sur deux grands volets :
- L'interface graphique, qui gère l'affichage et l'interaction utilisateur.
- La partie fonctionnelle, qui réalise les traitements d'image sur les pixels.

2. Interface graphique.

L'interface a été développée à l'aide des composants uicontrol et figure de Scilab.
Elle a été pensée pour être ergonomique, claire, et surtout dynamique, afin de s'adapter à différentes tailles d'écran.

2.1 Structure générale.

L'application est divisée en deux zones principales :

Zone d'affichage principale :
C'est la partie gauche de la fenêtre, où l'image sélectionnée est affichée qui correspond à 3/4 de la fenêtre.
Elle est gérée par un axe graphique (image_axes) qui se met automatiquement à jour à chaque traitement appliqué.

Zone de commande latérale :
Placée sur la droite, elle contient tous les boutons permettant de manipuler l'image : chargement, exportation, seuillage, amélioration du contraste, histogramme, etc.

Si un redimensionnement est fait en changeant fig_scale, les proportions occupées par la zone graphique et le menu seront conservés.

2.2 Fenêtre dynamique et gestion du ratio.

L'interface est dynamique. Lors de son lancement, la taille initiale de la fenêtre est définie, mais tous les éléments sont ensuite ajustés automatiquement selon le ratio de mise à l'échelle, noté fig_scale.

La variable fig_scale contrôle la proportion des objets graphiques (boutons, zones, espacements, etc.) afin de garantir une mise en page homogène quel que soit le format d'écran.

Ainsi, si la fenêtre est agrandie ou réduite, tous les composants s'adaptent de manière fluide sans déformation ni chevauchement.
Il est à noter que le redimensionnement de la fenêtre a été bloqué (pour des raisons d'esthétisme), et que fig_scale sert à ajuster la taille de la fenêtre manuellement (par défaut fig_scale = 0.8, et doit être inférieur ou égale à 0.9).

2.3 Menu et boutons.

Le panneau de commande regroupe deux sections distinctes :

La gestion des images qui permet de :

- Charger une image : Si aucune image est chargé, affiche l'image par défaut.
- Chercher une autre image : ouvre une boîte de dialogue pour importer une nouvelle image depuis l'explorateur de fichiers de l'ordinateur, il faut ensuite la charger dans la zone image grâce au bouton charger.
- Exporter l'image : nomme l'image actuellement affichée avec un nom unique basé sur le nom du fichier de base, concaténé à _modified concaténé à un timestamp. Puis enregistré dans le dossier source de l'image.

Ainsi qu'un mini menu dédié aux traitements d'image, totalisant 14 traitements lié à un bouton unique :

- Noir et Blanc
- Seuillage
- Amélioration du contraste
- Rotation
- Flip horizontale
- Flip verticale
- Addition d'image
- Histogramme
- Convolution (flou ou détection de contours)
- Eclaircissement
- Assombrissement
- Négatif
- Sobel (détection de contours)
- Posterisation (réduction du nombre de niveaux)
- Vieillissement / Effet sépia

Les boutons sont regroupés dans un mini-menu avec un système de défilement vertical (Monter / Descendre) généré dynamiquement.

Chaque bouton est lié à une fonction spécifique grâce à un callback, ce qui rend l'interface interactive : un clic déclenche immédiatement le traitement correspondant.
Afin d'améliorer la compréhension de la fonction de chaque bouton, une infobulle est visible si le curseur survole un bouton.

3. Partie fonctionnelle : traitement d'image.

La partie fonctionnelle correspond à l'ensemble des opérations appliquées à l'image chargée.

Toutes les fonctions utilisent la variable globale image_axes pour accéder à l'image actuellement affichée et la modifier directement dans l'interface.

3.1 Conversion en noir et blanc.

La fonction détecte si l'image est en couleur, puis applique une conversion en niveaux de gris avec rgb2gray().

Chaque pixel est ramené à une seule intensité entre 0 et 255.

3.2 Seuillage.

Le seuillage transforme l'image en binaire selon une valeur de seuil fixe (souvent 128).
Les pixels supérieurs deviennent blancs et les inférieurs noirs, ce qui met en évidence les formes principales.

3.3 Amélioration du contraste.

Le contraste est étiré sur toute la plage dynamique disponible.
Les intensités sont rééchelonnées de manière linéaire entre 0 et 255, améliorant la visibilité des détails sombres et clairs.

3.4 Rotation.

Tourner90() : effectue une rotation de 90° dans le sens horaire.

3.5 Flip/ Retournements.

FlipH() : retourne horizontalement l'image (effet miroir).

FlipV() : retourne verticalement l'image.

3.6 Addition d'image.

Cette opération additionne chaque pixel avec lui-même, ce qui augmente la luminosité globale. Par choix arbitraire, l'image est additionné avec elle-même. Cela fonctionne également si une autre image est chargé.

3.7 Histogramme.

La fonction crée une nouvelle fenêtre affichant l'histogramme de l'image actuelle.
Elle convertit d'abord l'image en niveaux de gris, puis trace la répartition des intensités à l'aide de histplot().
L'histogramme se met à jour selon les modifications de l'image.

3.8 Convolution.

La convolution applique un filtre 3×3 sur l'image pour effectuer un flou ou détecter les contours. Par exemple, un filtre de moyennage permet d'adoucir l'image, tandis qu'un filtre de Sobel peut révéler les bords. Ici le filtre, est un filtre permettant de flouter l'image.

3.9 Eclaircissement.

Augmente la luminosité de l'image de 20 unités, en gardant les valeurs dans l'intervalle [0,255].

3.10 Assombrissement.

Diminue la luminosité de l'image de 20 unités, en gardant les valeurs dans l'intervalle [0,255].

3.11 Négatif.

Inverse les couleurs de l'image : chaque pixel devient 255 - pixel, produisant un effet négatif.

3.12 Sobel.

Applique un filtre de détection de contours (Sobel) sur l'image convertie en niveaux de gris.

Il calcule la magnitude du gradient horizontal et vertical pour mettre en évidence les contours.

3.13 Posterisation.

Réduit le nombre de niveaux de gris de l'image (ici 4 niveaux) pour produire un effet cartoon.

3.14 Vieillissement.

Applique un effet sépia à l'image couleur, donnant un rendu "vieillissement" ou photo ancienne.

4. Conclusion

Ce projet illustre la complémentarité entre interface utilisateur et traitement numérique d’images sous Scilab.

L’interface graphique, entièrement dynamique grâce à la variable fig_scale, offre une présentation adaptable et ergonomique, tandis que les fonctions de traitement permettent d’explorer concrètement les transformations d’images de base.

Chaque opération visuelle est immédiatement observable, ce qui rend l’application à la fois pédagogique et interactive.