RAPPORT DU SYSTEME EXPLOITATION

Application Distribuée D'analyse D'images

Groupe C09

ZHENG Tao & GUO Xinrui

16/01/2015

Table des matières

I Introduction

Le but de ce projet est de construire un interprète de requêtes pour l'analyse d'images distant. Nous allons stocker les images en forma tga dans un répertoire dans la machine de serveur, et le client envoie le requête au serveur qu'il va le traiter et donner la réponse. Et puis, pour la fonction de serveur, nous allons réaliser de l'analyse des images, retourne l'histogramme de taux de rouge/bleu/vert; ensuite, nous montrons la répartition d'images selon leur taille. Enfin, nous réaliserons que le client peut envoyer des images au serveur, il peut mettre à jour des images même lorsque l'interprète ne tourne pas.

II La présentation des fonctions utiles

2.1 Fonction élémentaire

2.1.1 Convertir les espaces dans le socket et les reconvertir

Dans les expériences pratiques, nous trouvons que si nous envoyons les espaces dans le « socket », ça peut poser des problèmes. Afin d'éviter ces situations, nous remplaçons les espaces apparu dans les requetés avec les '<' en passant la fonction *ToStringHis* (pour les détaille, voir le ficher *strhelpers.c*), et dans la cote serveur, nous remplaçons les '<' apparu dans les arguments par les espaces en passant la fonction *convertirString* (pour les détails, voir le ficher *strhelpers.c*).

2.1.2 Tracer les histogrammes

En référant une méthode pose sur internet (voirhttp://www.google.fr/url?sa=t&rct=j&q=&esrc=s&source=web&cd=9&ved=0CFIQFjAI&url=http%3A%2F%2Fwww.tjhsst.edu%2F~dhyatt%2Fsuperap%2Fcode%2Ftarga.c&ei=ahGrVNbnDNavacH1gOAO&usg=AFQjCNHJofOVDERIfuoXSFD0DbXxg0uQGA&sig2=ADgZi1tFtGe0wp59lwbE9A&bvm=bv.82001339,d.d2s), nous utilisons les fonctions « *drawHis_size* », « *drawHis_threeRect* » et « *drawHisImage_main* » de tracer les histogrammes (pour les détails, voir le ficher *drawHistogramme*) cette méthode nous aide de tracer une représentation graphique avec un format de « tga ».

Pour tous les requêtes, qui demande une représentation graphique, le serveur va envoyer une image vers le cote client, et les images sont stocke dans

../client_recv_image_from_server

2.1.3 Fonction « help »

Dans notre code, nous ajoutons une requête « help », qui permet de aider les utilisateurs et de les montrer les requêtes canoniques, et les informations et les fonctions sur les requêtes que les utilisateurs peuvent être intéresse.

2.1.4 Fonction « quitter »

La fonction « quitter » va permettre d'arrêter le programme tous de suite.

Commande : quitter

2.2 Fonction principale :

Dans la cote client, afin de laisser l'ordinateur être plus habituelle vers les hommes, ici nous utilisons une fonction *readInMyWay* (pour les détails, voir le ficher *ReadWriteInMyWay.c*) qui va lire la requête dans la cote client, cette fonction nous permettre enlever les espaces automatiquement devant ou arrière la requête.

2.2.1 Créer la banque d'image :

Au tout début, afin de réaliser tous qui nous voulons faire, par exemple, montre l'histogramme de chaque couleur d'une image ou donne une représentation graphique qui montre la répartition des images selon leur tailles, il faut d'abord créer la banque d'image, c'est-à-dire créer une liste qui contient tous les informations essentielles (par exemple, le nom d'image ou la taille d'image, pour les détails, voir la définition de ***struct NodeImage*** ficher ***CreateBankImage.h***). Afin de créer la liste, il faut rappeler la fonction ***readBankImage*** qui prend un argument du chemin du répertoire, et elle permet de créer une liste qui contient tous les images, à la fin cette fonction retourne le premier nœud si la liste a été bien créé, sinon elle va reporter l'erreur. Pour rappeler cette fonction, dans la cote client, il faut taper « acquerir » afin de lancer cette opération.

Commande : acquerir

2.2.2 Afficher les informations des images dans un répertoire spécifique :

Avant d'afficher les informations des images, il faut assurer que nous avons déjà créé la banque d'image (autrement dit, la liste a été bien créée, sinon il va retourner une erreur).

Si la création a été bien fait, dans la cote client, il faut taper « afficher » pour lancer cette opération. Cette requête va nous permettre d'afficher tous les noms des images.

Commande : afficher

2.2.3 Analyser une image et donner les histogrammes selon les options :

Pour calculer l'histogramme, nous utilisons un tableau en dimension 2 qui stocker les taux de la présence de chaque chanel, ici nous considérons que his[0][n] (d'où 0<=n<256) concerne la chanel bleu, his[1][n] (d'où 0<=n<256) concerne la chanel verte, his[2][n] (d'où 0<=n<256) concerne la chanel rouge.

Afin de demander l'opération de tracer l'histogramme, la requête dans la cote client suive un format canonique :

histogramme « image_name » « option »

Dans notre cas, il y a 4 options principales :

histogramme ImageName r (ou « R », ici nous ne distinguons pas la majuscule et le minuscule)

Cette commande sert à calculer l'histogramme qui montre le taux de rouge.

histogramme ImageName g (ou « G », ici nous ne distinguons pas la majuscule et le minuscule)

Cette commande sert à calculer l'histogramme qui montre le taux de vert.

histogramme ImageName b (ou « B », ici nous ne distinguons pas la majuscule et le minuscule)

Cette commande sert à calculer l'histogramme qui montre le taux de bleu.

histogramme ImageName bgr (ou « BGR », ici nous ne distinguons pas la majuscule et le minuscule)

Cette commande sert à calculer l'histogramme qui montre le taux de bleu, vert et rouge dans la même figure.

2.2.4 Analyse et tracer l'histogramme représentant la préséance de 3 couleur

Dans les espaces RGB, si nous avons connu les valeurs dans chaque chanel d'un pixel, ce n'est pas toujours facile de trouver couleur, donc afin de juger le pixel soit en rouge, soin en vert, ou soit en bleu, nous changeons l'espace RGB vers l'espace HSV, d'où la valeur de H représente la teinte, qui est plus simple d'identifier le couleur. Avec la formule ci-dessous, nous passons changer les espaces :

$$V_{max} \leftarrow max(R,G,B)$$
$$V_{min} \leftarrow min(R,G,B)$$
$$L \leftarrow \frac{V_{max}+V_{min}}{2}$$
$$S \leftarrow \begin{cases} \frac{V_{max}-V_{min}}{V_{max}+V_{min}} & \text{if } L < 0.5 \\ \frac{V_{max}-V_{min}}{2-(V_{max}+V_{min})} & \text{if } L \geq 0.5 \end{cases}$$
$$H \leftarrow \begin{cases} 60(G-B)/S & \text{if } V_{max} = R \\ 120+60(B-R)/S & \text{if } V_{max} = G \\ 240+60(R-G)/S & \text{if } V_{max} = B \end{cases}$$

if $H < 0$ then $H \leftarrow H + 360$ On output $0 \leq L \leq 1$, $0 \leq S \leq 1$, $0 \leq H \leq 360$.

La fonction *identifyColor* (pour les détails, voir le ficher *AnalyseImage.c*) qui nous permet de réaliser les formule ci-dessus.

En référant le site sur internet (http://mkweb.bcgsc.ca/color_summarizer/?faq) qui décrit les ranges de chaque couleur, nous considérons que les ranges de chaque couleur (c'est-à-dire rouge, vert et bleu) sont données ci-dessous :

0<=h<90 && 330<=h<360----------rouge

90<=h<210----------vert

210<=h<330----------bleu

histogramme ImageName rrr (ou « RRR, ggg, GGG, bbb, BBB », ici la commande sera 3 chiffre en même et ainsi nous ne distinguons pas la majuscule et le minuscule)

Cette commande montre que les taux des présences de chaque couleur (RGB) dans une image en même temps.

Commande : histogramme « image_name » « option » (r/g/b/rgb/rrr, pour tous les commandes, voir la partie ci-dessus)

2.2.5 Analyser un répertoire d'images

Pour repartir le répertoire d'images selon les différentes options, principalement il y a deux options, un est selon la taille d'une image et l'autre selon le couleur en majoritaire.

2.2.5.1 Repartir les images selon leurs tailles

Pour la répartition selon les tailles d'images, dans la cote client, le requête doit suivi le format comme ci-dessous :

Classifier size « argument (seul les chiffre sont autorisé, le nombre d'argument est <10) »

Cette commande nous permet de repartir les images selon leurs tailles, en fait dans la cote client, les requêtes sont reçus comme un type de « String », mais dans la cote serveur, nous avons besoin d'un type de « int », donc nous utilisons la fonction *argStringToNb* (pour les détails, voir le ficher *strhelpers.c*).

L'algorithme de repartir les images selon leur tailles est :

1. Mise en ordres des arguments en augmentant (qui construisent les intervalles de diviser les images selon les tailles en ordre)
2. Parcours la liste créée par la banque d'image, mets les images dans les bons intervalles.

2.2.5.2 Repartir les images selon leurs couleurs en majoritaire

classifer couleur « couleur(r/g/b ou R/G/B) »

Le principale est même que la partie **Analyse et tracer l'histogramme représentant la préséance de 3 couleur**, en aidant la fonction *identifyColor* (pour les détails, voir le ficher *AnalyseImage.c*), nous pouvons savoir quel couleur est en majoritaire dans une image, et puis, parcours toute la liste, nous pouvons obtenir le résultat (pour les détaille, voir la fonction *divisercolor* dans le ficher *AnalyseRepertoire.c*).

Commande : classifier size/color « 10 30 (seul les chiffre sont autorisé, le nombre d'argument est <10) / r (g/b)»

2.2.6 Envoyer une image de la cote client vers la cote serveur

Envoyer « imageName » (ici le imageName est le nom complet, c'est-à-dire avec le chemin complet, soit vous pouvez mettre l'image dans le même répertoire du code, soit vous pouvez ajouter le chemin absolu devant le nom d'image)

Quand la cote serveur a reçu l'image, le serveur va automatiquement mettre à jour de la liste.

Commande : envoyer imageName

Remarque :

1. En fait, nous ne savons pas la taille limite d'une image de l'envoyer vers le serveur, mais nous avons sur que la taille doit moins que 20M (qui nous avons déjà testé)
2. Dans certain système, surtout pour les systèmes Ubuntu, quand nous avons envoyé les images de la cote client vers la cote serveur, souvent ça pose des problèmes de droits d'ouvrir les images, par contre, tous sont bien marchés sur les ordinateurs à l'école.

III Conclusion

Nous avons réussi de résoudre les analyses d'images en donnant les histogrammes qui représentent les taux de la présence de chaque couleur et aussi tracer les 3 rectangles qui montrent les taux de chaque couleur dans une image. Nous avons ressui les analyses de répertoire, afin de repartir le répertoire selon les tailles d'images ou les couleurs en majoritaire. Et nous pouvons envoyer les images de la cote client vers la cote serveur sans des problèmes avec une image qui n'a pas une grande taille.