Promo 2019

Rapport de soutenance 1

Ismail Keskas, Dan Azoulay,

Melvyn Petrochy, Lucas Dessert

Novembre 2015

Nom du groupe : Schrödinger

Nom du projet : RTFPicture

Chef de projet : Keskas Ismail (iso)

Table des matières

Introduction

Ce document est le rapport de la premiere soutenance. Il a pour but de présenter une synthse sur le travail fourni par l'équipe en charge du projet. Cette quipe est formée de quatre étudiants en deuxime année du cycle préparatoire de l'EPITA : Dan AZOULAY (azoul_d), Melvyn PETROCHY (petro_m), Ismail KESKAS (keska_i) et Lucas DESSERT (desser_l).

Notre projet est un logiciel de reconnaissance facial codé en C99 sous Linux, qui sera capable de rechercher des personnes présentes sur une photo donné, et qui partir de cette photo donnera la liste des personnes présentes sur la photo qui auront été préalablement enregistré dans la base de donnée du logiciel.

1 Présentation du groupe Schrödinger et répartition des taches

1.1 Origine du groupe

Nous étions tous les quatre éleves dans la meme classe l'année precedente en SUP. Nous avons décidé de former cette équipe pour mener a bien ce projet informatique d'envergure car nécessitant une organisation solide entre les différents membres.

1.2 Répartition des taches

	Ismail	Dan	Melvyn	Lucas
Détection d'un visage sur une photo	★	★		
Traitement de l'image			★	★
Base de données			★	★

2 RTFPicture

2.1 Fonctionnement

L'utilisation du logiciel est assez simple. Tout d'abord l'utilisateur devra remplir sa base de données de visages et nommer les personnes ajoutées pour que le logiciel puisse ensuite traiter ces photos, en extraire les informations qui lui seront utiles pour une futur reconnaissance, et la sauvegarder dans sa base de donnée.

Une fois ceci fait, l'utilisateur peux ensuite ajouter une photo quelconque au logiciel qui l'analysera et lui dira quels sont les personnes qu'il a reconnue en croisant les visages détectés sur la photo et sa base de données. Le logiciel sera aussi capable de signaler les absents, et proposera a l'utilisateur d'ajouter les nouveaux visages détecté (s'il y en) dans sa base de donne pour pouvoir les reconnaitre la prochaine fois.

L'utilisateur pourra a tous moment gérer la base de données, en ajoutant ou supprimant des personnes sa guise (Il pourra bien sr modifier le nom des personnes déj enregistré s'il le souhaite).

3 Détection de visage

3.1 Niveau de gris et pixels de l'image

Pour réaliser la reconnaissance faciale, il fallait dans un premier temps traiter l'image en niveau de gris, ce qui permet de faciliter les calculs. Pour cela, nous avons crit un algorithme parcourant une image pixel par pixel. Chaque pixel de couleur étant basé sur trois variables rouge,vert et bleu.

Pour obtenir l'image associe en niveau de gris, il suffit de faire la moyenne de ces trois variables puis de l'affecter chacune de ces variables. En effet, le niveau de gris se caractrise par le fait que les variables rouges, vertes et bleues soient gales. Pour parcourir l'image pixel par pixel et modifier la couleur de ces derniers, nous avons utilisé des fonctions appartenant la bibliothque SDL telles que GetRGB pour rcuprer les valeur rouge verte et bleu du pixel associé, SDL MapRGB qui elle permet par exemple de modifier la couleur d'un pixel... Cette fonction prend en paramtre une SDL Surface et sauvegarde le résultat dans une autre image grace a la fonction SaveBMP(surface, om de l'image.

3.2 Les caractéristiques pseudo-Haar

Pour la détection du visage, nous avons implementé la méthode présente par les chercheurs Paul Viola et Michael Jones. Leur methode permet la detection d'objets présents sur une image numérique, en particulier pour détecter la présence de visages sur une image.

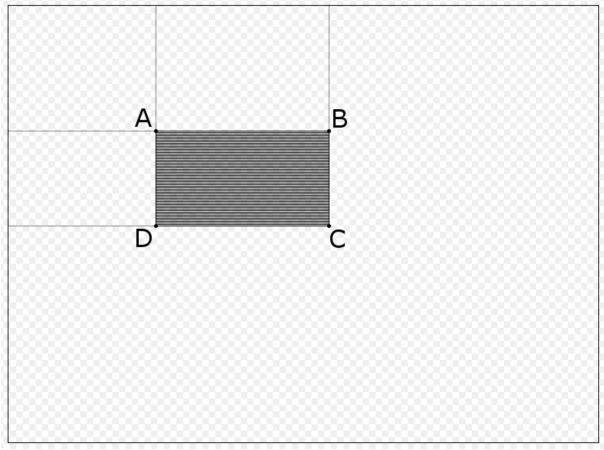
La méthode consiste a parcourir l'ensemble de l'image pixel par pixel tout en calculant un certain nombre de caractéristiques afin de déceler ou non la présence d'un ou plusieurs visages sur celle-ci. Cependant, l'étude pixel par pixel peut s'avérer longue et lourde en terme de travail du processeur. C'est ainsi que l'utilisation des caracteristiques de Haar est introduite. Ces caracteristiques sont des zones rectangulaires chevauchent chacune de quelques a plusieurs centaines de pixels.

De plus, nous effectuons séparement la somme des différents pixels présent dans la zone blanche du rectangle et ceux dans la zone noire. Nous calculons ensuite la différence des deux valeurs précedement calcules. En effet, nous parcourons totalement une premiere fois l'image ayant une certaine taille, avec une petite fenetre au depart mesurant 24*24 pixels et nous y appliquons a l'intérieur de celle-ci les différentes formes de Haar. Nous obtenus un certain nombre de résultats issus de nos différents calculs dans la fenetre 24*24 pixels, que nous stockons dans une file car facilitant l'insertion rapide de nouvelles données.

Ainsi, nous réiterons le procédé a nouveau sur la meme image mais cette fois-ci en augmentant la taille de notre fenetre se déplacant sur l'image et ce jusqu'a ce que la fenetre face la taille de l'image. Au final, les caracteristiques seront calculées a toutes les positions de l'image permettant donc la détection d'un ou plusieurs visages a n'importe quelle position sur l'image. En outre, nous avons utilisé cinq formes différentes de Haar afin d'augmenter la précision des calculs et donc l'effacite de détection des visages par notre logiciel.

3.3 Image integrale

Les images intégrales sont utilisées pour accélerer le calcul des caractéristiques pseudo-Haar. Il s'agit d'une image construite a partir de l'image d'origine dont chaque pixel de l'image est égal a la somme des pixels situes au-dessus et a gauche de celui-ci.



Ainsi, si l'on souhaite calculer la somme des pixels présent dans la zone grisée, il suffit de récuperer le résultat du calcul A+C-D-B, avec A,B,C et D des pixels contenant la somme de tous les pixels présent a gauche et au-dessus de ces derniers. Cette méthode accélere considérablement le calcul de la somme des différents pixels présent dans une zone et permettre ensuite d'utiliser ces résultats pour le calcul des caractéristiques de Haar.

3.4 Adaboost

Pour mettre en place l'algorithme d'apprentissage d'adaboost, il a fallu d'abord récupérer des exemples d'images positives et ngatives au format 24x24. Par la suite, nous avons créé une structure Example qui est composé de deux entiers correspondants au label (1 ou -1 selon positif ou négatif) et au poids (initialement,1/2*(nbimagespositives),respectivement,1/2*(nbimagesnegative)) et d'une file contenant toutes les caractristiques de l'image associe. Ainsi,dans un premier algorithme, nous avons créé un tableau d'Example qui regroupe tous les poids, labels et caractristiques des images fournies en exemples.

Par la suite, nous avons mis en place un second algorithme qui se charge de comparer les iemes caractéristiques de n images dntranement. En effet, cet algorithme prend en parametre le tableau d'exemple créé précédemment un entier j correspondant au caractéristique sur lequel est basée la comparaison et un entier n correspondant au nombre d'images dntranement. Avant dxécuter cet algorithme pour un j donné, il faut avant tout trier le tableau d'Example en ordre croissant en fonction du j ieme caractéristique de chaque image. Ensuite, en comparant le poids des images, le label et la valeur du caractéristique des différentes images, on arrive a déterminer une regle appelée aussi stump associé au caractéristique j. Cette regle est composée d'un seuil, d'un label, d'une marge et d'un taux d'erreur.

Chaque regle obtenue est considérée alors comme un classifier faible qui peut alors etre évaluée. En effet ,notre troisieme fonction, la fonction d'évaluation va en fonction des différents éléments fournis par la regle du caractéristique retourner 1 lorsqulle suppose qu'il y a un visage et -1 dans le cas contraire.
Sachant qu'il y a 162336 caractéristiques de Haar dans une image de 24x24 cela veut dire qu'il y a un tres grand nombre de regle,et ce serait extremement long de toutes les valuer. C'est pourquoi nous avons mit en place notre quatrime algorithme BestStump qui lui va permettre de choisir le meilleur classifieur, qui aura le plus petit taux d'erreur. Cet algorithme compare les différentes regles et conserve celle qui est la plus efficace.
Enfin llgorithme d'adaboost en lui meme, consiste a rappeler la fonction BestTump un T fois et de mettre ensuite a jour les poids afin d'obtenir un classifieur fort composé de la somme de ces T classifieurs faibles multiplier eux mme par des coefficient. Nos classifieurs faibles sont crits dans un ficher StrongClassifieur.txt. Une fois le classifieur fort obtenu il ne reste plus qu'a évaluer sur une image les différents classifieurs faibles qui composent le classifieur fort. Si le résultat est négatif on passe a la fenetre suivante sinon on stocke les coordonnées de la fenetre associée.

4 Base de donnée

4.1 Sa structure

Affin de grer principalement l'ajout et la suppression des visages dans le logiciel, il nous fallait crer une base de donn adapt a nos besoin. Elle doit pouvoir enregistrer une nouvelle personne, c'est a dire son nom, une ou plusieurs photos, et stocker les rsultats du pr-traitement de l'image pour seront utiliss pour la reconnaissance de la personne.

Au dpart nous avons rflchis comment crer cette base de donne, et quoi utiliser pour y arriver. Nous avons tudier les possibilits de la bibliothque SQL par exemple, qui est l'une des plus connus pour les bases de donns, mais nous avons finalement choisi de n'utiliser aucune bibliothque particulire et de coder nous mme en C notre propre structure de donn. Nous n'tions pas convaincue par l'utilit de SQL pour notre projet et il nous a paru plus simple de crer notre propre type de donn affin de l'adapter parfaitement au besoin, et de la moduler plus facilement. De plus, aucuns d'entre nous n'avait des connaissances (mme basique) dans la SQL.

Nous avons cod notre base de donnes sous forme de liste dynamique chane, pour ne pas avoir de problme de taille maximum ou d'allocation mmoire trop petite.Chaque personne ajout se retrouve tre un lment d'une liste chan avec un pointeur sur llment suivant de la liste (s'il existe). Chaque lment de cette liste dispose de trois string, l'un pour le nom de la personne qui est enregistr dans cet lment, et les deux autres seront interprt comme des pointeurs sur un dossier, le premier pointeur pointera sur le dossier qui contiendra le ou les images de la personne, et le deuxime pointera sur le dossier dans le quel les rsultats des traitements de l'image seront enregistr (dans des fichier .text). L'avantage de cette structure est qu'au lancement de du logiciel, un simple parcourt d'un fichier texte qui contient tous les noms enregistr permettra de reconstituer la liste dynamique reprsentant tous les visage que le logiciel est capable de reconnatre.

4.2 Au demarrage

Mais pour russis reconstituer cette liste chan nous avons nomm les trois string de chaque lment d'une faon spcifique. Par exemple, si on enregistre une personne sous le nom de Xavier, la string du nom sera videment "Xavier", la string qui fait office de pointeur sur le dossier qui contient la ou les photos de Xavier contiendra "Xavier-Image" et la troisime string qui permettra de trouver le dossier ou sont enregistr les rsultats du pr-traitement de l'image contiendra "Xavier-Character". De cette faon un simple fichier text qui contient tous les noms des personnes enregistr (un registre) permettra via une fonction que nous avons cod, de reconstituer la liste dynamique tel qu'elle tait lors de la dernire activit du logiciel.

4.3 L'ajout

Pour ajouter un lment a notre liste chan, nous avons cod une fonction qui prend en paramtre la liste chan ainsi que le nom de la personne a ajouter et qui vas donc crer un nouvel lment, remplir les trois string en fonction du nom de la personne, et l'ajouter la tte de liste (car l'ajout en fin de liste impliquerai de parcourir toute la liste avant de procder l'ajout).

En plus de a, la fonction d'ajout va crire dans le fichier texte qui sert de registre, le nom de cette personne, affin qu'elle puisse tre aussi ajouter dans la liste chan au prochain dmarrage du logiciel.

Nous avons aussi cod une fonction qui permet d'ajouter plusieurs lment d'un coup, elle prend en argument la liste dynamique ainsi qu'un tableau de string.

4.4 La suppression

La fonction de suppression elle, recherche llment supprimer au sein de la liste dynamique, le supprime si elle le trouve et efface son nom dans le registre ainsi que les deux dossiers qui contiennent les photos et les rsultats du pr-traitement de l'image.

4.5 La modification

Notre fonction de modification permet pour le moment de modifier le nom d'une personne dj enregistr dans la base de donn (elle modifie l'lment dans la liste dynamique ainsi que le nom crit dans le registre).

5 Les prochains objectifs

5.1 Reconnaissance de visage

Pour la soutenance finale, nous prvoyons d'implmenter une interface graphique simple et intuitive qui permettra via de simple boutons, de visualiser les photos des personnes dj enregistr, de supprimer ou d'ajouter des personnes dans la base de donne, et bien videmment de lancer une recherche de personne prsente sur une photo.

5.2 L'interface graphique

5.3 l'optimisation

6 Conclusion

7 Annexes