Le mémoire de Master E.P.H.E SVT de
Cognition Naturelle et Artificielle 2010

smileD : Sourire Naturel et Sourire Artificiel.

De l'utilisation d'OpenCV pour le tracking, la reconnaissance des expressions faciales et la détection du sourire

contenant:

Daniel Devatman Hromada

sous la direction de **Charles Tijus**

Lutin Userlab
Cité des sciences et de l'industrie

Respectables Professeurs et Maîtres d'École, Mesdames, Messieurs,

Le texte que vous tenez dans vos mains est le mémoire qui présente le travail effectué par l'étudiant Daniel Devatman Hromada durant le stage au Laboratoire des Usages en Technologie d'Information Numérique (Lutin) lors des troisième et quatrième semestres de ses études de Cognition Naturelle et Artificielle en vue de l'obtention du diplôme de Master 2 des Sciences de la Vie et de la Terre de l'École Pratique des Hautes Etudes.

Le sujet de stage initialement défini concernait « **L'imitation des expressions faciales par le visage robotique Roboto** ». Mais le problème s'est vite avéré tellement complexe qu'il a été impossible de faire entrer la totalité des idées et travaux dans le cadre d'un mémoire classique, ayant la forme d'un article scientifique. Il a donc paru plus raisonnable d'exploiter le savoir-faire obtenu lors du première semestre d'études à l'EPHE grâce à l'U.E. “Communication écrite” et de rédiger chacune des trois parties principales du texte selon un plan différent :

La première partie mets en œuvre le plan de rédaction OPERA (Observation, Problème, Expérimentation, Résultats, Action) afin 1) d'introduire au lecteur le visage robotique Roboto – qui peut être considéré comme le facteur déclencheur de tout ce que est présenté ci-dessous et 2) de présenter au lecteur la bibliothèque OpenCV. La deuxième partie mets en œuvre le plan de rédaction ILPIA (Introduction, Littérature, Problème, Implication, Avenir) pour présenter non seulement les expériences effectuées à la fin du première semestre (dont le but était d'exploiter les contours pour créer un système robuste et rapide de reconnaissance des expressions faciales) mais surtout pour faire entrer le lecteur dans le domaine fascinant de l'apprentissage automatique (*machine learning*).

Mais c'est, en effet, la troisième partie où se trouve le point culminant du travail : après un certain échec dû au problème de généralisation vers les échantillons inconnus qui termine le deuxième chapitre, l'auteur s'est décidé à réduire ses objectifs de S4 à un seul objectif plus réaliste: la problème de **reconnaissance des sourires.** C'est dans ce chapitre final que le plan de rédaction le plus courant en sciences cognitives, celui de IMRED (Introduction, Méthode, Résultats et Discussion), fut mis en œuvre. Le résultat final du travail - l'article *Semi-supervised haartraining of a fast&frugal face-coupled open source haarcascade detector of zygomatic smiles* – a été placé en annexe puisqu'il ne nous fut pas permis d’écrire le mémoire proprement dit dans une langue autre que le Français. Le texte se termine par la proposition du stage qui à l'origine des ces travaux.

Abstract

Three different approaches were examined in order to establish the facial-expression-based communication channel between robotic face of Roboto and its human counterpart. We had started with a point-tracking system based upon Lucas-Kanade optical flux algorithm. The need for a calibration phase as well as the fact that this approach is a purely behaviorist (i.e. stimulus-reflex based) one, without any cognitive representation of the "smile" on the side of the computer induced us to implement some more robust machine learning techniques. Therefore, as a second trial, we have studied the feasibility of a facial expression recognition system based on contour extraction, chamfer matching and subsequent feature selection by means of AdaBoost. While the initial tests limited to JAFFE dataset have shown promising results, generalisation to other datasets have shown to be problematic. Lastly, a "classical" approach of cascades of boosted haar-feature-based classifiers was applied upon SMILEs dataset and coupled with already existing face detectors, producing relatively fast (< 1ms per frame) and sufficiently accurate (90.5% AUC when tested on JAFFE) zygomatic smile detector (smileD). Both smileD and SMILEs dataset are published hereby as a gift to an OpenCV community.

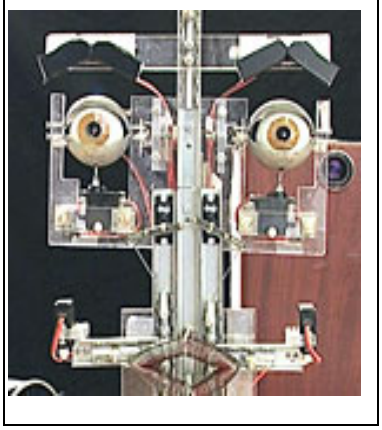
Résumé

Trois approches différentes furent examinées avec pour l'objectif l'établissement d'un canal de communication basé sur les expressions faciales entre le visage robotique Roboto et l'utilisateur humain. Les travaux débutèrent par les études du fonctionnement du système de suivi des points, basé sur l'algorithme de flux optique de Lucas-Kanade. La nécessité de la phase de calibration, aussi bien que le fait qu'il s'agit d'une approche purement «béhavioriste », i.e. basé sur le couplage stimulus-réflexe où aucune représentation interne du sourire n'est jamais présent sur le côté ordinateur, nous a fait passer à des techniques plus évoluées d'apprentissage automatique. C'est pourquoi nous avons ensuite étudié la faisabilité d'un système de reconnaissance d'expressions faciales grâce à l'extraction des contours, matching de *chamfer* et une sélection des traits effectuée ultériérement par AdaBoost. Même si les tests initiaux limités à la base d'images JAFFE donnèrent des résultats encourageants, la généralisation par rapport à d'autres bases d'images s'est avérée problématique. Enfin, l'approche "classique" exploitant les cascades de classifieurs de Haar a été mise à l'épreuve. Les résultats - i.e. un détecteur des sourires (*smileD*) rapide (< 1ms) et suffisamment précis (90.5% AUC quand testé sur JAFFE) aussi bien que la base d'images *SMILEs* utilisée pour générer ce dernière – sont offerts à la communauté OpenCV.

1. L'imitation par suivi des points

Observation

Figure 1 : Roboto

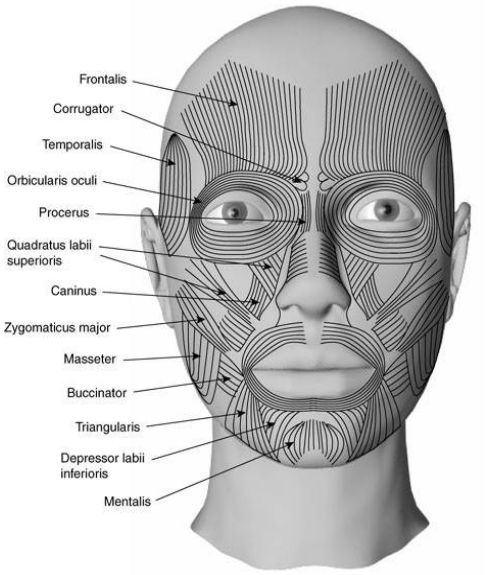


Roboto est une tête robotique qui a été conçue en 2003-4 dans une collaboration CNRS entre l'équipe développementale de Jacqueline Nadel (CNRS UMR7593, Centre Emotion) et l'équipe neurocybernétique ETIS de Philippe Gaussier, inspirées par la conception initiale de Feelix par l'équipe de Lola Canamero (Canamero & Fredslund, 2001). . Le maître d'œuvre fut Pierre Canet (Centre Emotion) . La tête est reliée à un ordinateur portable via une connexion série RS 232 reliée à une carte (SSC-12 de Lynxmotion). Douze servo-moteurs (Channel Serial Servo Controller) commandés à partir d'un programme élaboré par l'équipe ETIS permettent de faire bouger les sourcils, les paupières, et la bouche, en composant 6 expressions émotionnelles (tristesse – joie – surprise – colère- peur et neutre), formatées en référence aux unités d'action des expressions prototypiques humaines (Ekman et al. 2002) .

Tableau 1 : Les attributes des diverses moteurs de Roboto

Moteur	Point d'attache	Mouvement	Muscle associé
0	Coin de lèvre	horizontale	Zygomaticus major
1,3	Narine	verticale	Quadratus lab. sup.
2	Coin de lèvre	horizontale	Zygomaticus major
4	Menton	verticale	Depressor lab. inf. / Mentalis
5,6	Paupière	verticale	Orbicularis occuli
7,1	Sourcil int.	quasiverticale	Corrugator
8,11	Sourcil ext.	quasiverticale	Temporalis
9	Front	verticale	Frontalis

Figure 2 : Les muscles du visage



Etant donné que le contrôleur SSSC permet de gérer 12 moteurs et que nous en disposions de 14, il nous fallait choisir les deux moteurs qui ne pourraient pas être connectés. Ce sont les moteurs permettant de bouger les yeux autour de leur axe vertical que nous avons débranchés et ainsi exclus de nos expériences.

Ce choix fut partiellement motivé par le fait que les caméras qui étaient installées originalement dans les deux yeux du robot ne démontrèrent que de très faibles performances. Il fallut, donc, trouver une nouvelle solution pour rendre Roboto plus apte à voir. Après quelques

tentatives d'utiliser les cameras de la gamme la plus élevée (modèle : Pointgrey, mode de connexion: Firewire) comme la voie d'entrée des données perceptives, ce fut enfin une camera web assez simple (modèle : PHILIPHS, mode de connexion : USB), attachée sur le locus entre et un peu au-dessus les yeux (la position « du troisième oeil») qui fournit le flux d'images au système.

Le controlleur SSC-12 qui assurait la communication entre l'ordinateur et les servomoteurs fonctionnait de la manière suivante: par la voie du port de série RS-232, il recevait de l'ordinateur les séquences de trois octets dont le première a toujours une valeur de 255 ; le deuxième octet spécifiait le numéro du servomoteur (1-12) concerné aussi bien que la vitesse du mouvement (0-7 où 7 désigne la vitesse la plus rapide) ; le troisième octet spécifiait la position finale du servomoteur. Comme chaque octet peut avoir $2^8 = 255$ valeurs, chaque servomoteur peut se retrouver dans 255 états. Sachant que Roboto a 12 degrés de liberté, le nombre des ses états possibles est alors 12^{255} . Comme la différence entre les états voisins des servomoteurs est invisible pour un observateur humaine (e.g. on peut pas distinguer la différence entre l'oeil fermé dû au fait que le servomoteur 5 est en position 23 et celui dont le servomoteur est en position 24 ou 25), le nombre des états possibles du Roboto qui pourraient être réellement perceptibles en tant que les états spécifiques par un sujet humain est sans doute moins élevé que 12^{255} mais reste cependant si grand que même après plusieurs mois de travail, Roboto est toujours capable de nous étonner voire amuser par une expression faciale jamais remarquée auparavant.

Selon le célèbre FACS (Facial Action Coding System) (Ekman & Friesen 1977), il existe 7 expressions faciales de base qui sont repérables dans toutes les cultures du monde: 1) neutre 2) joie 3) surprise 4) tristesse 5) colère 6) peur 7) dégoût. FACS fut repris comme la référence de base par les concepteurs du Roboto: «Beaucoup d'efforts furent investis afin que les expressions du robot soient cohérents avec le système FACS» (Nadel et al. 2006). Les résultats de cet effort furent 6 séquences, l'une pour chaque expression faciale[1]. Chaque séquence se compose de 36 octets - 3 octets pour chacun de douze servomoteurs qui sont envoyés par la voie du port série au servo-controlleur SSC-12 pour produire l'expression demandée.

Expression	Séquence correspondante
Neutre	FF 10 86 FF 11 7C FF 12 7B FF 13 8A FF 14 5F FF 15 7D FF 16 8C FF 17 75 FF 18 85 FF 19 78 FF 1A 75 FF 1B 87
Joie	FF 10 86 FF 11 7C FF 12 7B FF 13 8A FF 14 5F FF 15 7D FF 16 8C FF 17 75 FF 18 85 FF 19 78 FF 1A 75 FF 1B 87
Surprise	FF 20 8C FF 21 76 FF 22 60 FF 23 93 FF 24 AD FF 25 5A FF 26 64 FF 17 60 FF 18 A1 FF 29 AA FF 1A 60 FF 1B 99
Tristesse	FF 10 3C FF 11 3C FF 12 C8 FF 13 CF FF 14 46 FF 15 87 FF 16 96 FF 17 95 FF 18 9F FF 19 46 FF 1A 95 FF 1B 9F
Colère	FF 10 86 FF 11 7C FF 12 7B FF 13 8A FF 14 3C FF 15 87 FF 16 96 FF 17 60 FF 18 80 FF 19 3C FF 1A 64 FF 1B 83
Peur	FF 20 5F FF 21 72 FF 22 A5 FF 23 94 FF 24 5A FF 25 23 FF 26 32 FF 27 80 FF 28 85 FF 29 AA FF 2A 80 FF 2B 85
Dégoût	FF 30 86 FF 31 BB FF 32 E4 FF 33 B4 FF 34 89 FF 35 7D FF 36 8C FF37 4B FF 38 85 FF 39 4E FF 3A 75 FF 3B 9C

Tableau 2 : Les expressions faciales de Roboto et leurs séquences d'octets correspondantes

1 Avec l'exception de la séquence pour dégôut que nous n'avons pas trouvée parmi les données fournies avec Roboto. Nous devions, donc, de “trouver” la séquence pour dégôut nous-même et nous l'avons fait grâce au script ri.pl qui sera présenté sur la page 7.

Le logiciel Docklight fut fourni avec les séquences comme le moyen d'envoyer les commandes au robot par le porte série. En d'autres termes, les 6 séquences + un logiciel pour Windows furent tout le «software» qui nous permettait de communiquer avec Roboto lorsqu'il fut transféré au (Lutin) en juin 2009.

Problème

En termes plus généraux, notre problème peut être formulé ainsi : comment rendre Roboto utile à notre laboratoire et à la communauté scientifique, voire utile au progrès des connaissances ?

Le fait que Roboto ne ressemble que peu au visage humaine peut aussi être considéré comme un problème. En effet - il n'y a pas de la peau, le cadre métallique est dans un seul plan et le robot donne donc une impression 2-dimensionnelle dont la partie la plus réaliste est la bouche, les lèvres étant formées d'une bobine qui se dilate ou se contracte selon le positionnement des moteurs 0,1,2,3,4. En effet beaucoup de ceux qui sont entrés en contact avec Roboto lors de leurs visites au Lutin ont remarqué que le robot est « peu réaliste », surtout par rapport aux visages robotiques comme « Einstein » (Wu et al. 2009) ou « Repliee Q2 Actroid ».

Mais il arrive que ce qui est considéré par le plus grand nombre comme un défaut se révèle être un avantage – il suffit de donner la priorité à une perspective plus optimiste. Tel était, est et sera notre démarche, et nous soutenons cette « approche optimiste » par les arguments suivants:

1) **L'effet de vallée dérangeante** : Il s'agit d'une réaction psychologique devant certains robots humanoïdes, d'abord suggérée par (Mori 1970) . Il décrit le fait que plus un robot humanoïde est similaire à un être humain, plus ses imperfections nous paraissent monstrueuses. Ainsi, certains observateurs seront plus à l'aise en face d'un robot clairement artificiel que devant un robot doté d'une peau, de vêtements et d'un visage pouvant passer pour humain. La Fig. 3 montre ce que nous pensons d'être la position de Roboto dans le schème de familiarité proposé par (Mori 1970). Un certain manque d'anthropomorphisme le met devant le premier «peak» et empêche ainsi les observateurs humains confrontés au robot de ressentir la chute négative

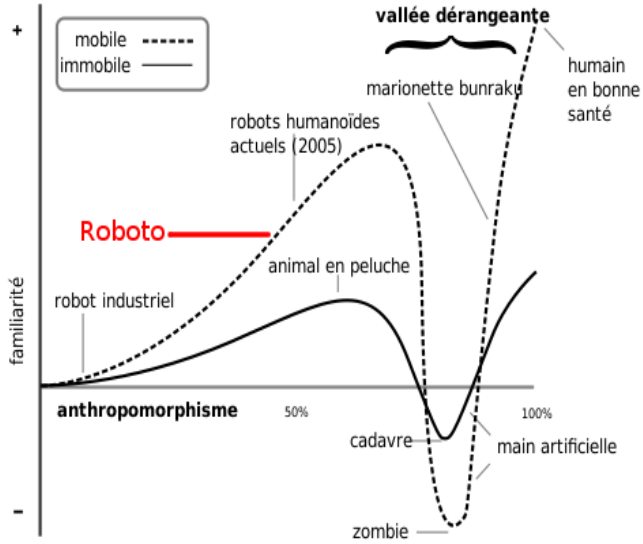


Figure 3 : La position de Roboto dans la schème antropomorphisme/familiarité de (Mori 1970)

dans la vallée de l'«inquiétante étrangeté» (Freud 1947)

2) **L'hypothèse des enfants autistes** – Comme l'a remarqué la mère du projet de Roboto, prof. Jacqueline Nadel, un certain manque d'anthropomorphisme était « fait exprès » afin de réaliser les expériences auprès d'enfants autistes qui ont souvent du mal à regarder un visage humaine. Selon son hypothèse , il pourrait en être autrement dans le cas d'interaction avec un visage à l'anthropomorphisme limité, comme celui de Roboto.

```
use Device::SerialPort;
my $neutre="FF 10 86 FF 11 7C FF 12 7B FF 13 8A FF 14 5F FF
15 7D FF 16 8C FF 17 75 FF 18 85 FF 19 78 FF 1A 75 FF 1B 87 ";
my $colere="FF 10 86 FF 11 7C FF 12 7B FF 13 8A FF 14 3C  FF
15 87 FF 16 96 FF 17 60 FF 18 80 FF 19 3C FF 1A 64 FF 1B 83 ";
my $sourire="FF 10 2F FF 11 D8 FF 12 D8 FF 13 1E FF 14 8C FF
15 7D FF 16 8C FF 17 75 FF 18 85 FF 19 78 FF 1A 75 FF 1B 85 ";
my $peur="FF 20 5F FF 21 72 FF 22 A5 FF 23 94 FF 24 5A FF 25
23 FF 26 32 FF 27 80 FF 28 85 FF 29 AA FF 2A 80 FF 2B 85 ";
my $tristesse="FF 10 3C FF 11 3C FF 12 C8 FF 13 CF FF 14 46 FF
15 87 FF 16 96 FF 17 95 FF 18 9F FF 19 46 FF 1A 95 FF 1B 9F ";
my $surprise="FF 20 8C FF 21 76 FF 22 60 FF 23 93 FF 24 AD FF
25 5A FF 26 64 FF 17 60 FF 18 A1 FF 29 AA FF 1A 60 FF 1B 99 ";
my $dat=""; my $command="neutre"; print ">";
my $port=Device::SerialPort->new("/dev/ttyS0");
$port->baudrate(9600);
while ($command=<STDIN>) {
  $port->lookclear;
  chomp $command;
  my $instruction=""; my $raw=0;
  if ($command eq "smile" or $command eq "sourire") {
    $instruction=$sourire;
  }   elsif ($command eq "neutre" or $command eq "neutral")
{
    $instruction=$neutre;
  }   elsif ($command eq "colere" or ($command eq "anger")
or ($command eq "angry")) {
    $instruction=$colere;
  } elsif ($command eq "peur" or $command eq "fear") {
    $instruction=$peur;
  } elsif ($command eq "tristesse" or $command eq "sad") {
    $instruction=$tristesse;
  } elsif ($command eq "surprise") {
    $instruction=$surprise;
  } else {
    $raw=1;
    my @d=split(/ /,$command);
    foreach my $s (@d) {
      $dat.=chr($s);
    } }
    if (!$raw) {
    my $i=0;
    while ($instruction=~/([A-F0-9]{2}) /g) {
       ++$i;
       print hex($1).",";
       $dat.=chr(hex($1));
    }
  }  print "\n>";
  $port->write($dat);
}
```

Code 1 : La code source du script PERL roboto.pl

Après avoir pris en compte ces deux arguments, nous avons décidé de ne pas interpréter le déficit d'anthropomorphisme de Roboto comme un obstacle, mais au contraire comme un avantage, comme une source des contraintes marquants le territoire de notre travail.

Or, Roboto fut accueilli au Lutin avec le but originel de rendre possible et facile les expériences concernant la problématique du traitement des expressions faciales par les enfants autistes. Et même si l'on peut dire que les travaux qui seront présentés dans les pages suivantes se sont partialement éloignés de cet objectif originel vers des régions inconnues et imprévues - comme c'est, d'ailleurs, souvent le cas en science - même si nos travaux se sont approchés de plus en plus du domaine d'un ingénieur de l'intelligence artificielle, nous considérons comme important de souligner le fait que nous n'avons jamais perdu de vue l'objectif expérimental voire médical (i.e. «aider»).

Au contraire, il se pourrait que ce que nous nous apprêtons à présenter ici ne soit qu'une introduction, un manuel expliquant le fonctionnement du Roboto à ceux de nos collègues qui se décideraient[2] à répondre la question : « L'interaction l'homme-machine par la voie des expressions faciales – pourrait-elle être exploitable pour les études aussi bien que pour **la thérapie** des troubles mentaux ? »

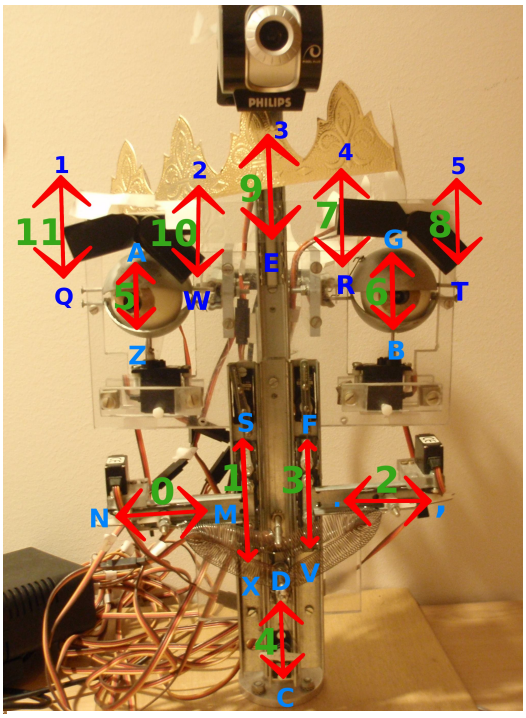
2 Les renseignements qui pourraient servir en particulier à un tel chercheu(r|se) seront dans le cadre de ce texte marqués par l'utilisation de police d'écriture souligné. Termes anglais sont écrit en *italique*.

Experimentation

L'objectif était alors bien défini : créer un logiciel qui traduit ce que le robot « voit » en son mouvement. Transformer 7 séquences de 36 octets associés aux 12 servomoteurs en outil expérimental également pour les études de l'autisme est un défi qui ne peut se résoudre que par un certain tâtonnement initial. Un défi d'autant plus difficile pour quelqu'un qui n'a jamais envoyé aucun octet vers un servocontrolleur, ni travaillé dans le domaine de la vision par ordinateur (computer vision), et tel était, en effet, notre cas quand ce défi fut relevé.

Mais il s'est avéré très vite, heureusement, que la majeure partie du travail était déjà faite non seulement par les concepteurs du Roboto, mais aussi par la communauté mondiale unie autour la philosophie de logiciels Open Source. Le module Device::SerialPort fut alors rapidement trouvé et choisi dans le répositoire du langage PERL (Wall & Loukides 2000) pour rendre possible la communication par le port série et servir ainsi comme la base de fonctionnement de notre premier script, appellé roboto.pl. Après son exécution, ce script roboto.pl attendit à l'entrée les mots clés. Si le mot clé comme « joie », « surprise », « peur », « colère », « dégout », «neutre » ou «tristesse » est écrit, le script envoie par le porte série la séquence des octets associés.

Figure 4 : Le couplage entre les moteurs de Roboto (numeros des moteurs en vert) et les touches (en blue) du clavier comme défini dans le script ri.pl



Les limites du script roboto.pl sont évidentes – le robot ne peut exprimer que l'une des 7 expressions préprogrammées: on ne peut pas accéder aux moteurs de manière individuelle, on ne peut pas régler la vitesse sans changer le script même. Pour franchir ces limitations, il fut conçu un deuxième script nommé ri.pl. Son fonctionnement était basé sur le couplage individuel entre les servomoteurs et le clavier. Pour dire la chose simplement, deux touches furent associées à chaque moteur, l'une pour augmenter l'octet qui représente la position du moteur (e.g. monter un sourcil), l'autre pour le faire descendre. Les paires touche/moteur furent choisies de telle manière que la position de touche dans le cadre du clavier QWERTY (c.f. Fig. 4) correspondît à la position du moteur par rapport à la totalité du visage (e.g. la touche 1 qui se trouve dans le coin en haut à gauche fait bouger le moteur du sourcil gauche extérieur vers le haut). Il y a donc 2x12 = 24 touches pour faire bouger les moteurs, une touche pour augmenter la vitesse, une pour la diminuer ; 6 touches *macro* pour les séquences de base et deux touches pour accélérer/décélérer la vitesse du mouvement.

Même si le script ri.pl nous donne la possibilité de travailler avec le Roboto de manière beaucoup plus « souple » que roboto.pl , il est évident que tous les deux ne représentent que le début du travail. En effet, les scripts nous permettent d'envoyer les commandes au robot pour le faire bouger, mais les données d'entrée, c'est-à-dire les données visuelles, ne sont jamais analysées par ces simples scripts. Autrement dit, le côté moteur de Roboto est mieux assuré qu'avant son arrivé au Lutin, mais le côté perceptif fait défaut. Jusqu'au ici, la machine ne *voit* rien.

La machine ne voit rien tant qu'on ne lui apprend pas à voir – tel est le postulat de base de la branche d'intelligence artificielle appellée « vision par ordinateur » (*computer vision* - CV). Depuis plusieurs décennies déjà, les chercheurs conçoivent des modèles théoriques, des formules mathématiques et des solutions de plus en plus évoluées pour effectuer le traitement et la classification d'images … pour aboutir enfin à OpenCV.

OpenCV est une bibliothèque écrite en langage C++, créée d'abord en filiale russe d'Intel pour ensuite devenir publique et *open source*. Il s'agit d'un projet qui contient centaines des fonctions inspirées par les études académiques portants sur tout ce que concerne la représentation numérique des données visuelles . Comme il n'est ni nécessaire ni possible de présenter, dans le cadre de ce mémoire, même un centième de tout dont OpenCV est capable, nous renvoyons le lecteur intéressé à l'ouvrage de Bradski & Kaehler (2008).

Une image n'est pour l'ordinateur qu'un tableau de pixels. Chaque pixel est un point coloré qui se trouve sur la position aux coordonnés X (la colonne) et Y (la ligne). Dans le cas d'un pixel coloré, la couleur est décrite par 3 entiers, un représente l'intensité du composant rouge, un du composant verte et un du composant bleu. Mais comme notre travail porte sur les expressions faciales et comme il est évident[3] qu'on peut reconnaître et classifier une expression faciale même sur une image en noir&blanc, on ne parlera que d'images aux pixels ayants un seul composant – celui de l'intensité (luminosité, la teinte de gris) sur l'échelle noir – blanc. Nous procédons ainsi car la réduction couleur → noir&blanc réduit sensiblement la complexité du problème à résoudre.

Nous répétons: une image de taille X x Y peut être décrite comme une matrice aux X colonnes et Y lignes dont chaque élément $pix_{X,Y}$ code l'intensité du pixel sur la position I(X,Y). Tandis que le rôle de la camera est de créer de telles représentations numériques, le rôle de l'ordinateur est de les traiter pour *en* tirer l'information digne d'intérêt, voire pour *y* trouver les objets de certaines catégories. Tout cela grâce à des procédés d'une essence purement mathématique, procédés que OpenCV dispense même à ceux qui ont du mal à comprendre les idées cachées derrière des termes tels: *un kernel de convolution, l'operateur de Laplace, le filter de Sobel* ou *la transformation de Fourier*.

Or, pour notre premier logiciel de traitement d'images – ou plutôt de séquences d'images, car

3 Du moins pour ceux qui se souviennent de l'époque où les films photos étaient en noir&blanc.

on parle ici du flux vidéo – nous nous sommes inspirés de l'appareil de « *eye tracking* » de FaceLab qui est présent au Lutin qui, lui aussi, est programmé grâce à OpenCV. Le principe de « eye tracking », ou même de tous les systèmes de suivi de points en général est le suivant: 1) repérer ou choisir le point d'intérêt au début de séquence vidéo ; 2) trouver les « coins » dans la proximité de ce point; 3) mettre en œuvre l'algorithme du flux optique pour trouver ces «coins» dans les images suivantes de séquence vidéo, en partant de la position repérée sur l'image précédente.

La notion de « coin » (*corner; feature*) est essentielle pour comprendre comment le système de suivi fonctionne. Un coin est un point d'image qui a des propriétés uniques par rapport aux autres points de la même image. Si on voulait suivre le point dans une vidéo d'un mur blanc, il serait assez difficile de repérer le même point dans l'image suivante car tous les points auraient à peu près la même intensité (seraient blancs). Au contraire, si on choisissait un point aux propriétés uniques, on pourrait le suivre, i.e. faire du *tracking*.

C'est justement ce que fait la fonction cvGoodFeaturesToTrack() de OpenCV. Elle est basée sur la définition de Harris & Stephens (1988) qui ne considère comme un coin que les points dont la dérivation dans les deux directions orthogonales est assez forte, ce qui revient, selon Shi & Tomasi (1994) à une valeur de dérivation[4] plus élevée qu'un certain seuil considéré comme un paramètre du système de tracking. Simplement dit : si la valeur d'intensité d'un pixel est assez différente par rapport à son voisinage tant en direction de l'axe X que de l'axe Y, ce point peut être considéré comme un coin, i.e. quelque chose d'unique, "un bon trait à suivre".

Une fois qu'on a repéré, dans la proximité du point d'intérêt, les points qui seront plus faciles à suivre que d'autres points, on peut analyser l'image suivante et essayer de les retrouver. Pour ce faire, il faut évaluer le mouvement qui fait différer les deux images. Les algorithmes du «flux optique» nous permettent de le faire. En raison de son élégance, de sa vitesse - il s'agit d'une méthode dit « creuse » (*sparse optical flow algorithm*) - et de son efficacité, nous avons choisi la méthode de Lucas-Kanade (LK) (Lucas & Kanade, 1981) pour calculer le flux entre deux images.

LK est basé sur 3 principes de base:

- la consistance de luminosité - l'intensité d'un pixel ne change pas entre les images suivantes
- la persistance temporelle – les changements entre images suivantes sont suffisamment lents
- la cohérence spatiale – les points voisins effectuent des mouvements similaires

En OpenCV, c'est la fonction cvCalcOpticalFlowPyrLK() qui nous permet de mettre en œuvre la méthode LK. Cette fonction ajoute aussi ce qu'on appelle « une pyramide d'images ». Une pyramide d'images est une représentation multi-résolution d'une image, créée à partir d'une image originelle – l'image originelle est sa base, le première étage est la base réduite à la moitié etc.

L'utilisation des pyramides d'images en combinaison avec l'algorithme L-K nous permets de

4 On peut aussi parler d'un "gradient"

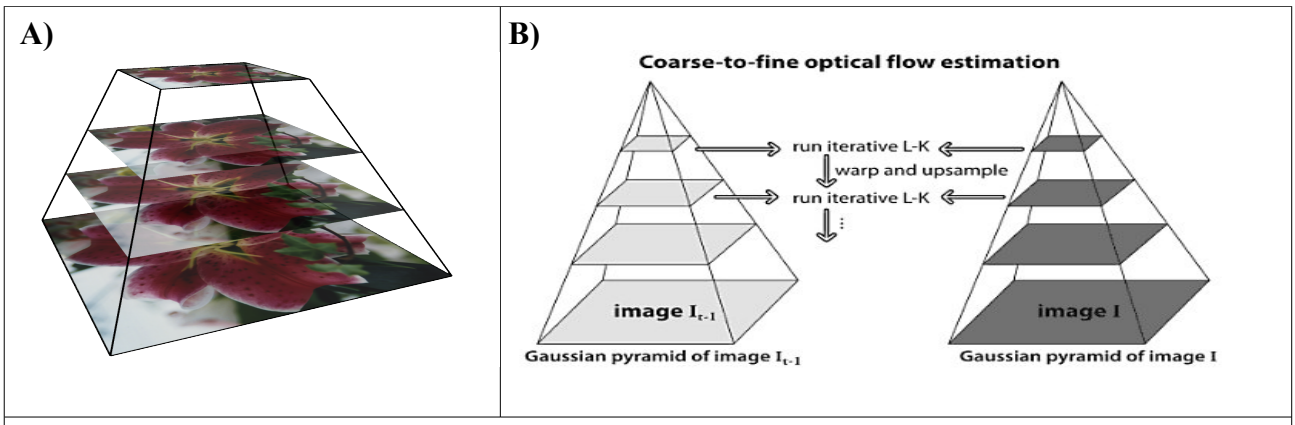


*Figure 5 : **A)** une pyramide d'images (l'image tirée de http://fr.wikipedia.org/wiki/Pyramide_(traitement_d%27image))*
***B)** utilisation des pyramides d'images dans l'estimation du flux optique de Lucas-Kanade afin d'atténuer les problèmes liées à la nécessité des petits mouvements (Bradski & Kaehler, 2008)*

réduire les limitations dues à la condition de « la persistance temporelle ». Un petit mouvement dans les hauts niveaux de la pyramide équivaut à un grand mouvement dans les niveaux inférieurs de la pyramide. Sans image pyramidale,ce dernier n'aurait pu être repérable par la méthode LK.

Pour récapituler : le «noyau perceptif» de notre premier logiciel d'imitation est le suivant : 1) l'utilisateur choisit le point d'intérêt par un clic de souris 2) la fonction cvGoodFeaturesToTrack() repère les coins dans la proximité du point d'intérêt (PoI), 3) elle retrouve les coins dans l'image suivante du flux vidéo grâce à la fonction cvCalcOpticalFlowPyrLK(), et elle en tire l'information sur la position nouvelle du PoI. Ceci peut être fait pour un nombre quelconque de points et la séquence qui est analysée, aussi bien que la vidéo où l'utilisateur choisit les PoIs, provient, bien sûr, de la caméra de Roboto.

Une fois la vision - le côté sensoriel du Roboto - mise en place, elle est couplée avec la motricité de la manière suivante : avant de faire le clic afin de choisir le PoI, l'expérimentateur appuie sous la touche de clavier faisant référence au moteur dont le mouvement sera couplé avec le mouvement de PoI sur lequel on s'apprête à cliquer.

Simplement dit, le moteur qui va bouger est défini par la touche du clavier (c.f. Figure 4) , le PoI qui sera suivi est défini par le clic de souris, et leur couplage est assuré par le fait que l'expérimentateur appuie sur la touche, puis fait le clic. L'envoie des séquences d'octets est ensuite analogique au script ri.pl, la plus grande différence étant que, cette fois-ci, nous mettons en œuvre la bibliothèque *libserial* de C++ et non *Device::SerialPort* de PERL car le logiciel *sm_imitation* que nous venons de présenter n'est pas écrit en PERL mais en C++ (pour être compatible avec OpenCV)

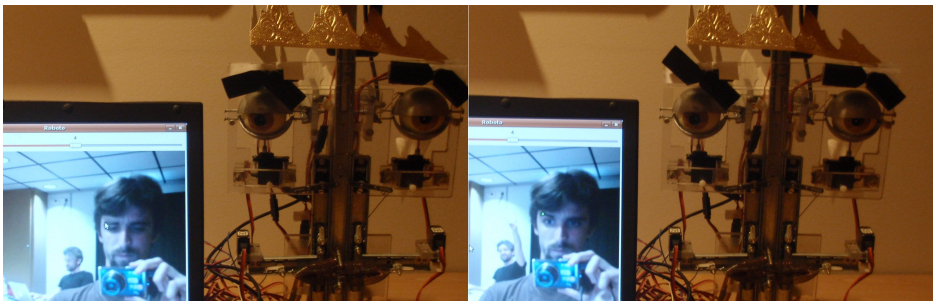


Figure 6 : L'imitation du mouvement du sourcil gauche effectué par le logiciel sm_imitation.c

Résultats

Il nous était et est difficile d'évaluer de manière fiable ce logiciel basé sur le suivi des points. Ceci est dû au fait que le logiciel ne fournit aucune sortie numérique, il ne fait que suivre des points et essayer de traduire le mouvement repéré dans le mouvement des moteurs (la procédure est illustrée sur Fig. 6). Le choix du point à suivre aussi bien que le choix du moteur qui sera couplé avec le PoI sont faits de manière *ad hoc* .

En d'autres termes, la phase de couplage moteur ↔ PoI, la phase de calibration ou l'intervention manuelle est nécessaire et doit précéder chaque expérience avec ce logiciel, ce qui rend chaque expérience unique, non répétable et donc non-scientifique.

Étant donné que nous ne voulons pas trop nous éloigner de nos objectifs scientifiques, contentons-nous alors de deux observations, l'une positive et l'autre négative, qui pourraient paraître superflues mais il n'en est rien. La bonne nouvelle est que la combinaison des fonctions cvGoodFeaturesToTrack() et cvCalcOpticalFlowPyrLK() permet même aux débutants en OpenCV de construire leurs premiers logiciels de suivi des points. La mauvaise nouvelle est que même si le système des images pyramidales est mis en oeuvre et même si l'algorithme de flux optique de LK est censé remplir la condition de persistance temporelle, le suivi robuste de points n'est pas assuré → on perd souvent les points surtout quand un mouvement brusque a été effectué, ce qui demande ensuite le recommencement de la phase de calibration.

Action

En termes concrets, il nous faudrait au moins 7 mois de plus pour rendre possible la mise en œuvre de l'imitation par le suivi des points (basé sur le flux optique de LK) si jamais nous décidions de mettre en oeuvre cette approche-là pour les expériences avec les enfants autistes.

Les phases de calibration, la nécessité de définir les meilleurs seuils des fonctions cvGoodFeaturesToTrack() et cvCalcOpticalFlowPyrLK(), l'incertitude de pertinence des résultats obtenus - nous ont amenés à quitter la technique « l'imitation par suivi des points » car il est peu probable qu'un enfant autiste passe dans un état d'âme paisible -sans mouvement brusque - la totalité de la phase initiale lors de laquelle il faudra coupler 12 points avec 12 touches → 12 moteurs.

Même quand on a essayé de tourner les yeux de Roboto, i.e. sa caméra, vers Roboto même, et faire Roboto suivre les mouvements de ses propres moteurs, la condition de la persistance temporelle n'était pas respectée et on a donc souvent perdu les points[5]. On imagine la difficullté d'une telle expérience quand il s'agit de la mener avec enfants autistes!

Mais le plus grand reproche que nous pouvons faire à l'approche présentée dans ce chapitre

5 Mais du temps en temps, quand le couplage était fait de manière particulière, il nous arriva ce qu'était prevu, i.e. Roboto bouga tout seule pour un certain moment sans aucune intervention humaine nécessaire

n'est pas d'ordre technique, mais d'ordre théorique. En effet, un logiciel qui ne fait que suivre les points et les traduit *bêtement* en octets envoyés aux servo-moteurs n'est qu'une solution *ad hoc*, un logiciel qui n'apporte rien ni au domaine de la psychologie cognitive, ni au domaine de l'intelligence artificielle qui nous attirait de plus en plus au fur et à mesure que nous continuions à connaître la bibliothèque OpenCV et la langue de programmation C++ qui nous étaient jadis inconnues.

Or, l'intelligence d'un tel système d'imitation par le suivi aveugle peut être comparé à l'intelligence d'une voiture de Braitenberg (Braitenberg, 1986) mais non à l'intelligence dite « émotionnelle » que les chercheurs en robotique émotionnelle tâchent de simuler et comprendre . Car dans le système de couplage point ↔ moteur que nous venons de présenter il n'y a que le système stimulus-réponse ; le logiciel donc relève plutôt de l'ordre béhavioriste que cognitiviste. Pour que le logiciel s'appele ainsi, il faudra autant que possible des niveaux de représentation différents, et y ajouter les aptitudes de classification, voire une sorte de généralisation.

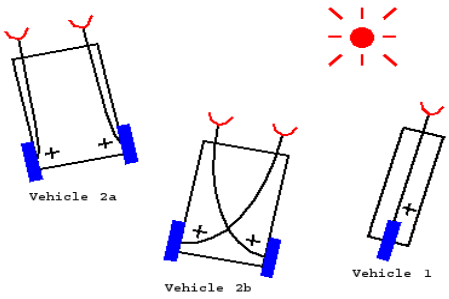


Figure 7 : L'exemple de couplage stimulus-action: Les voitures de Braitenberg. Chaque capteur sensible à la luminosité est couplé à une roue.

En un mot, il nous faudra de l'apprentissage.

Et pour faire cela, il nous faudra entrer dans le royaume de *machine learning*.

Bibliographie

Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media

Braitenberg, V. (1986). *Vehicles: Experiments in synthetic psychology*. The MIT press.

Canamero, L., & Fredslund, J. (2001). I show you how I like you-can you read it in my face?[robotics]. *IEEE Transactions on Systems, Man and Cybernetics, Part A*, *31*(5), 454–459.

Ekman, P., & Friesen, W. V. (1977). Manual for the facial action coding system. *Consulting Psychologist*.

Freud, S. (1947). Das Unheimliche (1919). *Gesammelte Werke*, *12*.

Harris, C., & Stephens, M. (1988). A combined corner and edge detector. Dans *Alvey vision conference* (Vol. 15).

Lucas, B. D., & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. Dans *International joint conference on artificial intelligence* (Vol. 3, p. 3).

Mori, M. (1970). The uncanny valley. *Energy*, *7*(4), 33–35.

Nadel, J., Simon, M., Canet, P., Soussignan, R., Blancard, P., Canamero, L., & Gaussier, P. (2006). Human responses to an expressive robot. Dans *Proceedings of the Sixth International Workshop on Epigenetic Robotics* (p. 79–86).

Shi, J., & Tomasi, C. (1994). Good features to track. Dans *1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94.* (p. 593–600).

Wall, L., & Loukides, M. (2000). *Programming perl*. O'Reilly & Associates, Inc. Sebastopol, CA, USA.

Wu, T., Butko, N. J., Ruvulo, P., Bartlett, M. S., & Movellan, J. R. (2009). Learning to Make Facial Expressions.

2. Introduction à l'apprentissage automatique en relation avec la reconnaissance des expressions faciales

Introduction

L'objectif à long-terme est de rendre possibles les expériences basées sur l'interaction entre Roboto et les enfants autistes. Comme c'est souvent le cas, les contraintes liées à cet objectif marquent la voie à suivre : sachant que le comportement des enfants en question varie souvent selon des conditions qu'on ne peut pas prévoir, nous avons décidé que l'appareil expérimental – et surtout le logiciel qui est son noyau – que nous tâchons de construire doit être aussi robuste, rapide et simple que possible, aussi bien pour l'expérimentateur que pour le sujet d'expérience. La contrainte de robustesse exige que le logiciel fasse ce qu'on lui demande de faire malgré des variations de conditions externes (différences de visages des différents sujets, différences de luminosité etc.). La contrainte de *vitesse* est due au fait que nous nous contentons d'un logiciel qui nous permet d'étudier l'interaction de haut niveau entre l'homme et la machine, et pour cela il faut que le logiciel nous permette de faire interagir l'homme et la machine en temps réel .

Enfin, quand nous parlons de *la simplicité*, nous voulons dire que nous préférons mettre en oeuvre une approche qui utilise un nombre minimal de paramètres à définir ou à régler lors de l'expérience. Dans le cas idéal il n'y en aura aucun, l'appareil étant complètement automatique et l'expérimentateur se contenant de démarrer le logiciel et d'analyser les résultats.

Même si l'approche présentée dans le cadre du premier chapitre était assez rapide, elle n'était ni robuste – on a souvent perdu les points suivis – ni simple – une phase de couplage assez longue aurait été nécessaire avant chaque expérience.

Or, nous avons décidé de tenter de construire un système dans lequel aucune phase de couplage ou de calibration préalable à l'expérience n'est nécessaire. Pour ce faire, il faut que le logiciel ait, avant l'expérience même, quelques connaissances des traits invariants des objets (les visages ou leurs expressions faciales) – qu'il va essayer de 1) reconnaître 2) classifier et 3) imiter.

Telle technique est possible si et seulement si le logiciel a des connaissances préalables et générales sur les objets à reconnaître. Le moyen par lequel on peut aboutir à des connaissances générales à partir d'un échantillon limité d'exemples concrets est appelé l'apprentissage, et la sous-discipline d'informatique qui étudie les algorithmes divers nous permettant d'effectuer l'apprentissage sur machines est connu sous le nom d'apprentissage automatique (*machine learning*). L'objectif de ce chapitre est d'introduire ces algorithmes et de définir la notion de «trait», tout cela en relation avec la reconnaissance des expressions faciales.

Littérature

Ce qu'on vise dans le cadre de la recherche en reconnaissance des expressions faciales (REF) est la classification des images selon les expressions faciales (EF) contenues dans les images analysées. En général, la classification s'accomplit en deux phases: 1) on extrait les traits (*features*) à partir d'une image ; 2)selon les traits extraits, et selon les couplages (traits - étiquette) fournis lors de la phase d'apprentissage, un algorithme d'apprentissage automatique (AA) attribue une étiquette même à un objet inconnu. L'étiquette désigne l'appartenance d'objet à une classe.

Les principes fondamentaux d'AA sont bien expliqués en (Bishop et al. 2006); l'ouvrage de (Haykin 1994) peut sans doute servir comme livre de référence. En ce qui concerne la mise en oeuvre d'AA dans le domaine de la vision par ordinateur (*computer vision – CV*), le 13ème chapitre de l'ouvrage de (Bradski & Kaehler 2008) pourra s'avérer des plus utiles, surtout pour les néophytes.

Quant aux divers algorithmes utilisés en AA, les plus connus sont : la distance de Mahalanobis (Mahalanobis 1936), *K-means* (Lloyd 1982), le classificateur de Bayes (naïf ou normal) (Minsky & Selfridge 1961), les arbres de décision (Breiman 1984), le *boosting* (Freund & Schapire 1996), les arbres aléatoires (Breiman 2001), la maximisation d'expectation (Dempster et al. 1977), les K voisins les plus proches (*K-nearest neighbors*) (Fix & Hodges Jr. 1989), les réseaux de neurones artificiels (i.e. les perceptrons multi-couches) (Rumelhart 1989) ou le SVM (*support vector machine*) (Vapnik et al. 1997). Chacun de ces algorithmes a des faiblesses et des points forts, ce qui est bien compris par le théorème TANSTAAFL[6] (Wolpert & Macready 1997) .

Tous sont inclus dans la bibliothèque ML qui fait partie intégrante d'OpenCV.

Quant à la REF, les années récentes en ont vu un véritable *explosion* dans ce domaine d'études. Le recueil de (Li & Jain 2005) et l'article de synthèse de (Fasel & Luettin 2003) en présentent les approches les plus courantes. Elles peuvent être divisées en trois groupes principaux: REF basée sur les ondelettes, REF basée sur les modèles et REF basée sur les contours.

Les ondelettes : A cause d'une certaine analogie avec le fonctionnement du cortex visuel (J. P. Jones & Palmer 1987), les ondelettes de Gabor (Lyons et al. 1998) sont souvent utilisées comme traits pour la REF. Même si cette approche semble être intéressante d'un point de vue théorique, ses exigences informatiques (dont sa lenteur) la rendent presque inexploitable pour la reconnaissance en temps réel (Azcarate et al. 2005). Au contraire, l'approche exploitant les traits rectangulaires ressemblant aux ondelettes de Haar (Lienhart & Maydt 2002) peut aboutir à des performances très intéressantes, surtout quand on la combine avec un certaine *astuce* d'images intégrales (Viola & M.

6 L'acronyme TANSTAAFL signifie *There Ain't No Such Thing As A Free Lunch* i.e. « il n'y a rien de tel qu'un repas gratuit »

Jones, 2002) et que l'on choisit l'algorithme AdaBoost pour déterminer les traits d'intérêt[7].

Les modèles: L'autre méthode pour aboutir à la REF est l'*Active Appearance Model* (AAM) proposé par (Cootes et al. 1998). Cette approche permet de construire un modèle statistique à partir d'un certain nombre d'exemples d'apprentissage. Le modèle ainsi construit – on peut l'imaginer comme une certaine grille ou masque – est ensuite apparié à une image (Abboud et al. 2004). Comme le système d'AAM est naturellement construit de telle façon qu'il prenne en compte la variabilité des objets auxquels le modèle sera apparié on peut profiter d'informations liées à cette variation pour faire l'apprentissage et la classification d'expressions faciales (Lucey et al. 2005). Même si cette approche mérite d'être suivie de très près, aucune solution open source, i.e. publique, n'existait pour l'AAM quand nous avons commencé notre stage[8].

Comme nous avons rapidement compris qu'essayer de reproduire le calcul matriciel de l'AAM serait hors de notre portée dans le cadre de notre stage de Master, nous nous sommes décidés à nous concentrer sur l'approche qui exploite **les contours.** L'approche que nous avons tentée de reproduire était celle de Moore et Bowden (2007)[9]

Problème

Le résumé d'article de Moore et Bowden (2007) propose :

« This paper introduces a novel method for facial expression recognition, by asssembling contour fragments as weak classifiers and boosting them to form a strong accurate classifier. Detection is fast as features are evaluated using an efficient lookup to a chamfer image, which weights the response of the feature. »

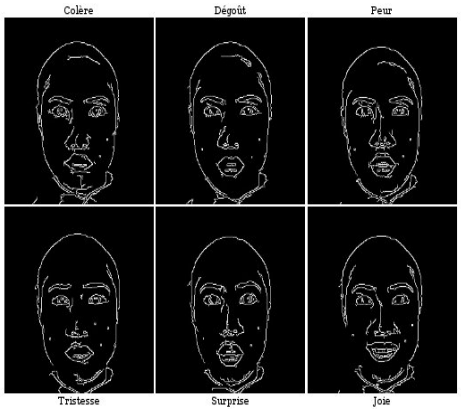


Figure 8 : Images transformées en contours obtenus grâce à la fonction cvCanny() sur 6 images exprimantes les EFs différentes

Expliquons d'abord les termes de base:

- **Contour** : courbe qui correspond au changement brutal de l'intensité lumineuse dans une image (i.e. la courbe est détectée là où se trouvent les grandes différences avec valeurs des pixels voisins). Pour repérer les contours on utilise souvent le filtre de Canny (Canny 1987) qui élimine beaucoup de faux contours puisqu'il ne cherche que les composantes connexes. La différenciation des contours se fait par seuillage à hysteresis nécessiant deux seuils, un haut et un bas, considérés en OpenCV commes les paramétres de la fonction *cvCanny*.

7 La théorie portant sur les traits de Haar sera expliquée au troisième chapitre de ce mémoire.
8 Entre temps, une solution *OpenCV positive* est apparue sur le site http://code.google.com/p/aam-opencv/
9 Il existe d'autres approches basées sur les contours, notamment celle appelée « edge oriented histograms » de (Dalal et al. 2006). Malheureusement nous ne fûmes informés de leur existence, par un expert d'Aldebaraan Robotics, que trop tard pour une étude en profondeur.

- **Fragment d'un contour** : morceau du contour coupé de manière aléatoire ; Une liste des points. Chaque fragment fournit un trait.

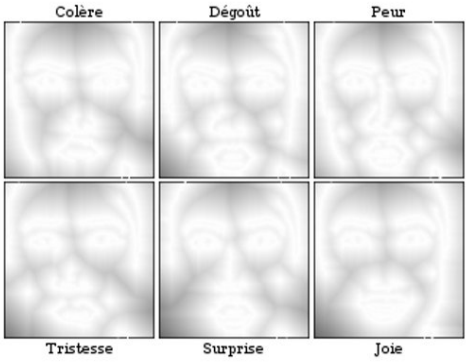


Figure 9: Les images de chamfer construites à partir des images d'origine (c.f. Fig. 8). L'intensité de chaque pixel dépend de la distance euclidienne du contour le plus proche sur l'image d'origine.

- **L'image de *chamfer* :** image dont la valeur $q_{x,y}$ du chaque pixel est proportionnelle à la distance au trait le plus proche (par rapport à X,Y) présent sur l'image d'origine. L'algorithme initialement proposé par (Barrow et al. 1978), nous publions sa version pour OpenCV en Annexe 2.

- **La distance de *chamfer*** : permet en théorie de repérer la ressemblance entre deux courbes. Dans le cadre de ces études, elle este plutôt considérée comme un moyen de numériser l'ampleur de la présence de la courbe C_1 tirée de l'image I_1 dans la même région que l'image I_2. Pour ce faire I_2 est transformée en image de *chamfer*, et la distance de chamfer n'est rien d'autre que la somme des valeurs des points d'image de *chamfer* I_2 qui ont les mêmes coordonnées que les points composant C_1. C'est comme si on faisait chevaucher par C_1 sur I_2 pour ensuite calculer la somme totale des valeurs de pixels sous C_1.

- **Classifieur faible :** également appelé « hypothèse » ; algorithme qui donne de meilleurs résultats que le hasard (i.e. il ne se trompe pas plus d'une fois sur deux en moyenne). Dans le cadre de cette étude, un classifieur faible est simplement un trait auquel est associé un arbre de décision n'ayant qu'un seuil de bifurcation (*stump*).

- **Classifieur fort** : combinaison linéaire des classifieurs faibles ; sortie d'algorithme de *boosting* ; résultat final de la phase d'apprentissage.

- **Les algorithmes de *boosting*** : groupe d'algorithmes de méta-apprentissage qui permettant de choisir, parmi un nombre très élevé d'hypothèses possibles, les hypothèses qui semblent être les plus « parlantes » et les plus « pertinentes » pour la classification finale.

- **AdaBoost** (*adaptive boosting*) : repose sur la sélection itérative de classifieurs faibles en fonction d'une distribution des exemples d'apprentissage. Chaque exemple est pondéré en fonction de la difficulté de sa classification par rapport au classifieur courant. (c.f. aussi Annexe 1)

Les notions de base de l'approche de Moore et Bowden ainsi expliquées, illustrons maintenant leur méthode sur un problème concret, celui de la REF dans les images appartenant à la base d'images Japanese Female Facial Expression (JAFFE) :

La JAFFE contient 204 images, et chaque image est labelisée par l'une de 6 étiquettes (la peur, la surprise, la joie, la colère, le dégoût ou la tristesse). La totalité des 204 images est divisée en 2 parties – l'échantillon d'apprentissage (*training sample*) ayant 177 images et l'échantillon d'essai (*testing sample*) 37 images. Le premier sera utilisé pour construire le classifieur fort, le deuxième sera utilisé pour voir comment le classifieur fort « se débrouille » avec les exemples qu'il n'a jamais rencontrés lors de la phase d'apprentissage.

Le processus d'apprentissage se déroule ainsi : toutes les images sont alignées, puis le filtre de Canny est mis en oeuvre pour obtenir une image binaire (un pixel est soit noir, soit blanc) représentant les contours connexes, ce qui réduit de manière significative la quantité de données et élimine les informations jugées moins pertinentes[10]. Chaque image est ensuite « inversée » (c.f. Annexe 2) à une image de *chamfer* dont les valeurs des pixels contiennent la distance vers le contour le plus proche. Pour chaque classe C des expressions faciales en question est pris un nombre T des morceaux des contours de 177/6=29 images appartenant à la classe C (ayant C pour étiquette). Puis, on fait comme si chaque des T fragments était placé sur chacune des 177 images de *chamfer* et on calcule la somme totale des valeurs de pixels chevauchées par le fragment en question. Le résultat est un nombre qui nous renseigne sur la distance du contour qui faisait partie d'une des images de classe C au contour le plus proche en image X.

En d'autres termes, pour chaque image X des 177 images on obtient ainsi un vecteur de T traits numériques dont quelques-uns, on l'espère, seront suffisants pour distinguer les images de la classe C des images apartenants à des classes différentes. Par exemple, disons que c'est la classe C=joie qui nous intéresse, et qu'on a décidé de n'utiliser que T=9 traits (i.e. fragments) pour la classification, on obtient alors 177 vecteurs (lignes) ayant 9 éléments chacun, e.g. :

232 , 324, 772 , 552, 923, 789, 87, 124, 87

984 , 398, 234 , 902, 892, 398, 56, 234, 12

etc.

En plus, comme on parle des images appartenant à la partie d'apprentissage, on connaît aussi l'étiquette de classe liée au vecteur – 1 si l'image exprime la joie, 0 si l'image n'y appartient pas. Si les traits sont extraits de manière pertinente, et si le nombre T n'est pas trop bas (en réalité, on utilise des centaines des milliers des traits), il est très probable que l'information contenue dans ces traits suffirait pour créer un classifieur – i.e. trouvera une sorte d'application[11] traits → étiquette capable de distinguer un visage joyeux de de tous les autres.

10 Tout en gardant intacte l'information nécessaire pour la détermination d'expressions faciales. Ceci est dû à l'hypothèse du départ (Moore & Bowden, 2007) : «l'information contenue dans les contours est suffisante pour la REF ». Nous référons à cette hypothèse comme à *une hypothèse de bande dessinée*.

11 Dans le sens mathématique du terme.

Les démarches pour créer un tel classifieur sont dans ce cas les suivants : pour chaque trait, c'est-à- dire pour chaque colonne dans notre matrice 177xT , nous cherchons un certain seuil – c'est-à-dire une certaine valeur numérique, qui différencie mieux les classes joie-absent ou joie-présent. On obtiendra alors T arbres de décision[12] qui nous serviront en tant que classifieurs faibles. Mais il y a peu d'espoir qu'il existe un seul trait, un seul contour capable de faire une distinction robuste entre la classe joie-absent et la classe joie-présente. Il paraît beaucoup plus convenable de supposer qu' une expression faciale est une combinaison de plusieurs traits. En d'autres termes, c'est en combinant les classifieurs faibles entre eux de manière adéquate qu'on peut espérer aboutir à une classification fiable.

L'algorithme AdaBoost sert justement à cela. Il combine N hypothèses faibles – qui sont dans notre cas les arbres de décision du type *si valeur < seuil → classe=X ; sinon → classe = non-X* pour trouver un classifieur fort. La condition pour les classifieurs faibles est simple : ils doivent classifier mieux que le hasard. S'il existe un nombre suffisant de tels classifieurs faibles, on peut être sûr que nous trouverons un classifieur fort classifiant sans erreur toutes les images d'apprentissage. Ceci est possible dans la mesure où l'algorithme est conçu de telle manière que le taux d'erreurs de classification des exemples d'apprentissage tombe exponentiellement vers zéro avec le nombre N des classifieurs faibles. Cette propriété mathématique d'AdaBoost est démontrée en (Freund et Schapire 1995)

Implication

Une fois que AdaBoost a choisi les traits « les plus parlants » et les a combinés de façon linéaire en construisant ainsi le classifieur fort, on peut mettre ce dernier à l'épreuve en le confrontant avec 37 images d'échantillons d'essai (*testing sample*) , c'est-à-dire avec les images qui n'étaient pas utilisées lors de la phase d'apprentissage.

	VP	VN	FP	FN
Colère	2	22	2	3
Dégoût	3	25	1	0
Peur	2	24	2	1
Tristesse	2	21	0	6
Surprise	4	23	0	2
Joie	6	20	0	3

Tableau 3: Les taux des vrai positifs, vrai négatifs, faux positifs et faux négatifs obtenus lors la classification de EF d'images de JAFFE

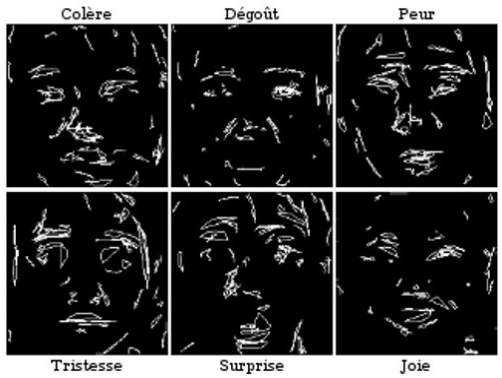
Les résultats d'essai sur JAFFE sont présents dans le Tableau 3 . Le terme « faux positif » (FP) fait référence à la situation où une image est reconnue comme contenant une EF tandis qu'en réalité il n'en contient pas. Au contraire le terme « faux négatif » (FN) fait référence à la situation où une image est reconnue comme ne contenant pas d'EF tandis qu'en réalité il en contient. Les termes « vrai positif » (VP) et « vrai négatif » (VN) font référence aux cas où l'algorithme a bien classifié la présence, ou bien l'absence d'une FE dans l'image analysée.

12 Dans ce cas, il s'agit d'arbres de décisions les plus simples, n'ayant qu'un seuil de bifurcation et alors de deux branches seulement (e.g. si la valeur du trait est plus grande que la valeur du seuil, le classifieur faible donne un vote positif pour l'appartenance à la classe d'intérêt et une vote negatif si jamais la valeur est moins grande)

Étant donné qu'il s'agissait de notre première tentative dans le domaine de l'apprentissage automatique, les résultats nous ont paru encourageants, surtout pour la classe « joie ». Nous nous sommes donc décidés à mettre à l'épreuve notre classifieur fort en le confrontant non avec des images venant d'une échantillon ayant les conditions de luminosité complètement différentes de JAFFE utilisée pour l'apprentissage.

Or, après avoir obtenu l'autorisation de la part de l'université Carnegie-Mellon d'accéder à la base d'images la plus reconnue dans la domaine de la REF, celle de Cohn-Kanade (Kanade et al. 2000), les résultats obtenus dépassèrent à peine le pur hasard.

Figure 10 : Fragments des contours (traits) choisis par AdaBoost comme les plus pertinentes pour déterminer la classe d'EF présente



Avenir

L'une des explications de cet échec peut résider dans le fait que l'approche présentée par Bowden et Moore ne fonctionne simplement pas aussi bien que le rapportent les auteurs, i.e. que les traits obtenus grâce à une image de *chamfer* ne sont pas pertinents pour la classification d'EF.

Même si les auteurs n'ont pas répondu à nos mails dans lesquels nous leur avons demandions quelques renseignements supplémentaires sur leur méthode, et même si leur méthode (en mai 2010) n'est reproduite nulle part dans la littérature, nous restons persuadés – et le Tableau 3 aussi bien que la Figure 10 l'indiquent - que la distance de *chamfer*[13] entre un contour et une image peut s'avérer un moyen très efficace pour la construction de traits suffisamment déterminants pour une classification basée sur les contours.

Or, nous expliquons le fait que nous n'avons pas réussi à classifier les images contenues dans la base Cohn-Kanade par les facteur suivants : 1) nous n'avons pas aligné les images selon un point de référence commun ; 2) nous n'avons appliqué aucune méthode pour remettre la luminosité des images au même niveau, ce qui pourrait être fait par paramétrisation du seuillage par hystéresis de filtre de Canny afin qu'il nous fournisse toujours environ la même quantité de contours. Nous croyons que le manque de ces deux facteurs a empêché de trouver un système de classification suffisamment général pour être appliqué même aux flux d'images venant de la caméra de Roboto.

De ce fait, non seulement nous avons compris que, pour faire face aux deux problèmes mentionnés dans le paragraphe précédent, il nous fallait plus que quelques semaines. Bien plus –

13 L'avantage de la méthode proposée ci-dessus est que, une fois calculée l'image de chamfer, la computation des valeurs des traits (sélectionnés préalablement par AdaBoost) est très rapide à effectuer. Qui plus est, on peut très bien imaginer que la phase de computation la plus exigeante – la construction d'une image de chamfer – peut se dérouler non sur un processeur central (CPU), mais pourra être renvoyée au processeur de la carte graphique (GPU). Le projet appellé OpenCL (à ne pas confondre avec OpenCV) nous paraît idéal pour atteindre cet objectif.

nous avons compris que pour construire un système de REF suffisamment robuste et rapide, il nous fallait, peut être, la période de toute une thèse, sinon un tel système aurait déjà été construit[14] .

En restant persuadés que *l'hypothèse bande dessinée* est vraie et que les contours peuvent suffire pour la REF, nous nous sommes décidés, enfin, à réduire nos objectifs et, en accord avec une ancienne maxime « moins est plus », nous nous sommes concentrés pleinement sur une seule EF – celle de sourire.

Bibliographie

Abboud, B., Davoine, F., & Dang, M. (2004). Facial expression recognition and synthesis based on an appearance model. *Signal Processing: Image Communication*, *19*(8), 723–740.

Azcarate, A., Hageloh, F., van de Sande, K., & Valenti, R. (2005). Automatic facial emotion recognition. *Universiteit van Amsterdam June*.

Barrow, H. G., Tenenbaum, J. M., Bolles, R. C., & Wolf, H. C. (1978). PARAMETRIC CORRESPONDENCEAND CHAMFERMATCHING: TWO NEW TECHNIQUES FOR IMAGE MATCHING. Dans *Proc. DARPA IU Workshop* (p. 21–27).

Bishop, C. M., & others. (2006). *Pattern recognition and machine learning*. Springer New York:

Bradski, G., & Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, Inc.

Breiman, L. (1984). *Classification and regression trees*. Chapman & Hall/CRC.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Canny, J. (1987). A computational approach to edge detection. *Readings in computer vision* , 184.

Cootes, T. F., Edwards, G. J., & Taylor, C. J. (1998). Active appearance models. *Computer Vision—ECCV'98*, 484.

Dalal, N., Triggs, B., & Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. *Computer Vision–ECCV 2006*, 428–441.

Dempster, A. P., Laird, N. M., Rubin, D. B., & others. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*(1), 1–38.

Fasel, B., & Luettin, J. (2003). Automatic facial expression analysis: a survey. *Pattern Recognition*, *36*(1), 259–275.

Fix, E., & Hodges Jr, J. L. (1989). Discriminatory analysis. Nonparametric discrimination: Consistency properties. *International Statistical Review/Revue Internationale de Statistique*, *57*(3), 238–247.

Freund, Y., & Schapire, R. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. Dans *Computational Learning Theory* (p. 23–37).

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. Dans *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-* (p. 148-156). Citeseer.

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR Upper Saddle River, NJ, USA.

Jones, J. P., & Palmer, L. A. (1987). An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, *58*(6), 1233.

Kanade, T., Tian, Y., & Cohn, J. F. (2000). Comprehensive database for facial expression analysis. *fg*, 46.

Lienhart, R., & Maydt, J. (2002). An extended set of haar-like features for rapid object detection. Dans *IEEE ICIP*.

Li, S. Z., & Jain, A. K. (2005). *Handbook of face recognition*. Citeseer.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137.

Lucey, S., Ashraf, A. B., & Cohn, J. (2005). Investigating spontaneous facial action recognition through AAM representations of the face. *Handbook of Face Recognition*, 275–286.

Lyons, M., Akamatsu, S., Kamachi, M., & Gyoba, J. (1998). Coding facial expressions with gabor wavelets. Dans *Proceedings of the 3rd. International Conference on Face & Gesture Recognition* (p. 200). IEEE Society.

Mahalanobis, P. C. (1936). On the generalized distance in statistics. Dans *Proceedings of the National Institute of Science, Calcutta* (Vol. 12, p. 49).

Minsky, M., & Selfridge, O. G. (1961). Learning in random nets. *Papers*, 335.

Moore, S., & Bowden, R. (2007). Automatic facial expression recognition using boosted discriminatory classifiers. *LECTURE NOTES IN COMPUTER SCIENCE*, *4778*, 71.

Rumelhart, D. E. (1989). The architecture of mind: A connectionist approach. *M*, *1*, 133–159.

Vapnik, V., Golowich, S. E., & Smola, A. (1997). Support vector method for function approximation, regression estimation, and signal processing. *Advances in Neural Information Processing Systems 9*.

Viola, P., & Jones, M. (2002). Robust real-time object detection. *International Journal of Computer Vision*, *57*(2).

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for search. IEEE Transactions on Evolutionary Computation, 1(1), 67–82.

14 Construit et publié en version *open source* par l'un de dizaines, voire centaines d'équipes de recherche qui, visent au même but en ce moment même !

3. L'évaluation des diverses versions du détecteur de sourire smileD par le jeu d'imitation avec le visage robotique Roboto

Introduction

Qu'est-ce qu'une expression faciale ? Selon (Ekman & Friesen 1977), une expression faciale est une combinaison d'unités d'action (UA), où chaque UA correspond au mouvement d'un des muscles faciaux. Les mêmes auteurs postulent, en accord avec (Darwin 1872), que de telles combinaisons d'UE existent et sont exprimées et interprétées de manière semblable dans tous les peuples du monde. Même si une telle universalité anthropologique des expressions faciales est remise en question par une étude récente de (Jack et al. 2009) pour les EFs au contenu affectif négatif, l'universalité de l'expression faciale au contenu positif *par excellence*, celle de sourire, n'en est rien concernée.

Contrairement à d'autres EF comme la peur, le dégoût ou la colère, le sourire dont on va parler dans le cadre de ce chapitre[15] est produit par le mouvement d'un seul muscle – celui du grand zygomatique (Ekman & Friesen 1982). Mais c'est ne pas seulement la simplicité et l'universalité du sourire qui nous amena à concentrer nos forces sur la construction d'un détecteur des sourire dès que nous nous sommes rendus compte que nous avions échoué dans la construction d'un système plus générale de REF.

Or, nous avons déjà dit que l'objectif ultime de notre stage était de construire un tel appareil expérimental pouvant servir des expériences, notamment dans le domaine du développement, ou de l'autisme. Concernant le développement, l'évidence que le sourire est l'un des premiers canaux de communication non verbale entre la mère et l'enfant (Strathearn et al. 2008). Quant à l'autisme, l'étude de (Dawson et al. 1990)a démontré que "*les enfants autistes répondent beaucoup moins aux sourires de leurs mères que les enfants normaux. En plus, on a trouvé que les mères des enfants autistes répondent beaucoup moins aux sourires de leurs enfants que les mères des enfants normaux*".

Serait-il alors possible que le sourire joue un rôle dans le développement de l'autisme ? En d'autres termes, serait-il possible ou envisageable que le défaut de communication mère ↔ enfant par la voie de sourire ne soit pas l'un des symptômes mais, peut être, l'une des causes de ce syndrome ?

Peut-être Roboto pourra-t-il nous fournir une réponse.

---

15 Précisons qu'un autre sourire existe aussi, appelé aussi le sourire de Duchenne (Duchenne de Bologne 1862), pour la production duquel la contraction des muscles autour les yeux i.e. *orbicuralii occuli* - est nécessaire .

La détection des sourires

Mais afin que Roboto puisse répondre à cette question, il faut l'équiper d'un détecteur de sourires (DS) . Quelques DS existent déjà, comme celui embarqué comme "smile shutter" dans les caméras Sony (Akita et al. 2010) ou ceux rapportés par (Deniz et al. 2008) ou (Whitehill et al. 2007).

Comme aucune de ces solution n'est une solution *open source*, il nous fallait tenter de construire notre propre DS. L'article de (Whitehill et al. 2007) nous était particulièrement utile pour choisir une méthode appropriée – les auteurs ont comparé plusieurs méthodes d'extraction des traits, c'est-à-dire : les histogrammes d'orientation de contours (c.f. note 9), les filtres de Gabor (c.f. page 14) et les filtres rectangulaires, le tout en relation avec deux méthodes d'AA (*SVM* et *boosting*).

C'était la combinaison filtres réctangulaires + *boosting* qui s'est averée la plus performante et la plus rapide. Ce qui était une bonne nouvelle car il s'est avéré que ces "filtres rectangulaires" ne sont rien d'autre que les caractéristiques de Haar, très bien integrées dans la bibliothèque OpenCV.

Les caractéristiques de Haar et l'image intégrale

Les caractéristiques de Haar (*Haar-like features* - HF) sont la matière brute de classifieur que nous apprêtons à présenter. Un classifieur basé sur HF ne classifie pas selon les intensités de pixels, mais selon les différences d'intensités entre deux, trois ou quatre régions rectangulaires de pixels. Une valeur numérique de HF résulte de l'addition ou de la soustraction des sommes d'intensités des régions rectangulaires des pixels.

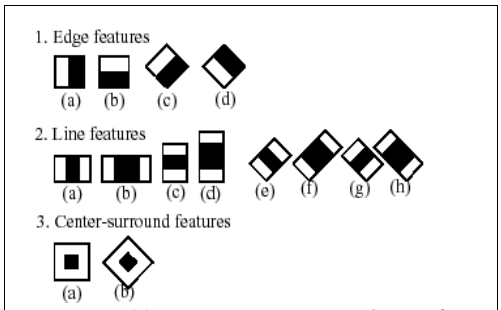


Figure 11 : HFs integrées dans OpenCV. Dans cette représentation des ondelettes, les régions blanches sont interpretées comme "ajouter cette région" et les régions noires comme "soustraire cette région" (Bradski & Kaehler, 2008) .

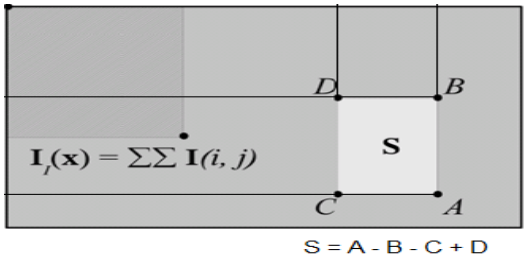


Figure 12. : Image intégrale I_i. La somme des pixels contenus dans le rectangle S (défini par les points A,B,C,D) peut être calculée en ne faisant que 4 réferences vers les points A, B, C, D d'image intégrale.

Ces valeurs d'intensité des régions rectangulaires peuvent être calculées très rapidement, une fois qu'on a construit ce que l'article époquale[16] de (Viola & Jones 2001) appelle "l'image intégrale". Une image intégrale *ii* contient sur les coordonnées x,y la somme des pixels de l'image d'origine qui se trouve au-dessus et à gauche de la position x, y :

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')$$

16 L'article cité plus que 3862 fois moins que 9 ans après son apparition est sans doute digne d'un tel adjectif.

En termes simples, l'image intégrale est une "astuce" mathématique qui permet de calculer, très rapidement, une fois qu'elle est construite[17], les sommes des valeurs des pixels dans les régions rectangulaires de l'image analysée.

La détection des visages selon la méthode de Viola et Jones

L'objectif de Viola et Jones était de construire un système robuste et rapide pour la détection des visages. Les deux contributions majeures - hormis la décision d'utiliser les HFs comme traits et accélérer le calcul de leur valeurs au moyen d'image intégrale - étaient: 1) d'utiliser AdaBoost comme moyen pour choisir les traits et construire les classifieurs ; 2) d'enchaîner les classifieurs dans une cascade attentionnelle.

Sélection des traits par AdaBoost: Sachant que le nombre des HFs possibles d'une image de 24x24 pixels est supérieure à 180 000[18], il aurait été impossible de le calculer pour toutes les sous-fenêtres de l'image où nous cherchons l'objet (le visage, le sourire) à détecter. Il faut donc choisir les traits, et ce que Viola&Jones furent les premiers à proposer de procéder par AdaBoost, après avoir proposé l'hypothèse que "un nombre très réduit de traits peut être combiné pour donner un classifieur efficace" .

La méthode que nous avons présentée dans le chapitre précedent était analogue à celle de Viola & Jones: nous cherchions les traits qui séparent le mieux les échantillons d'exemples négatifs et positifs. Pour chaque trait, l'algorithme d'apprentissage de classifieurs faibles tente de déterminer le seuil optimal pour la classification.

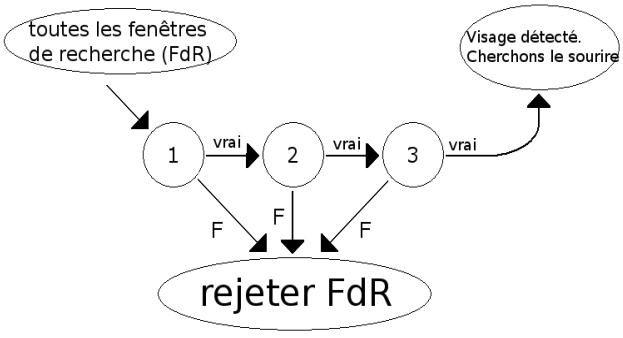


Figure 13 : La cascade de Viola & Jones

La cascade attentionnelle : Jusq'ici, la procédure de détection du visage comprend cette hiérarchie des représentations: *l'intensité des pixels → la somme des intensités des pixels dans une région rectangulaire → les différences entre plusieurs régions (HF) → les groupes de HFs sélectionnés par AdaBoost comme les plus pertinents pour la classification.* La contribution principale de Viola & Jones fut d'ajouter encore un niveau à cette hiérarchie computationnelle, d'organiser les groupes de HFs choisis dans les nœuds d'une cascade de réjection. L'idée de base est que, afin qu'une fenêtre de recherche (FdR) puisse être classifiée

17 L'avantage principal d'une image intégrale est qu'on n'a besoin que d'un seul passage à travers l'image d'origine pour la construire, en appliquant deux équations récurrentes: s(x,y) = s(x, y-1) + i(x,y) ; ii(x,y) = ii(x-1, y) + s(x,y). Pour comparer: la construction d'image de chamfer (c.f. Annexe 2), exige 2 passages par l'image d'origine (avant, puis arrière)

18 Notons, que pour une image, le nombre de traits de Haar possibles est beaucoup plus élevé que le nombre de ses pixels (24x24=576 << 180 000).

comme “contenant un visage”, elle doit être évaluée comme telle par tous les nœuds de la cascade. Au contraire, si jamais une FdR est classifiée comme “sans visage” par un noeud de la cascade quelconque, la FdR est d'emblée rejetée et le logiciel procède alors par l'analyse d'une nouvelle FdR. Après que toutes les FdR de l'image sont ainsi évaluées, la détection est terminée pour l'image donnée.

En plus, les nœuds de la cascade sont ordonnés de telle manière que les nœuds les plus rapides à évaluer (i.e. se composant de moins de HFs) sont mis tout au début de la cascade. Grâce à cela, un grand nombre de FdR sans visage est rejeté après l'évaluation d'un très petit nombre de HFs. Selon l'article de Viola & Jones, il faudra évaluer (en moyenne) 10 traits pour détecter un visage dans une FdR, ce qui oblige le processeur de regarder seulement 10 x (640-25) x (480 – 25) x 4 = 627300 fois dans la mémoire contenant la représentation d'une image intégrale pour trouver tous les visages de taille 25x25 pixels dans une image de 640x480 pixels.

Il s'agit donc d'une méthode tellement rapide qu'aujourd'hui on peut souvent voir les détecteurs des visages basés sur ce principe embarqués même dans les caméras numériques de moyenne gamme.

“Haartraining”

Même s'il y a beaucoup plus à dire sur le sujet, il serait peut-être, superflu de tâcher d''expliquer le *tour de force* de Viola & Jones plus en détail dans le cadre de ce mémoire. Nous renvoyons les lecteurs intéressés à l'article source (Viola & Jones 2001), ainsi qu'aux pages 506-516 de l'ouvrage de (Bradski & Kaehler 2008) où la méthode est mise en relation avec OpenCV.

Ce sont justement ces pages-là qui présentent le logiciel *haartraining*. Ce logiciel, qui fait partie intégrante d'OpenCV, automatise l'apprentissage des classifieurs basés sur la théorie de Viola & Jones. Comme ce sont les régions rectangulaires qui sont à la base de cette approche, *haartraining* permet d'entraîner les classifieurs – ou les détecteurs, car un détecteur n'est qu'un classifieur mis en œuvre dans toutes les FdRs - pour les objets composés de régions ou de blocs. Au contraire, il serait vain de tenter d'utiliser *haartraining* pour construire un classifieur capable de reconnaître les branches d'arbres.

Heureusement, un sourire zygomatique peut être considéré comme un objet composé de blocs.

SMILEs

Afin que *haartraining* puisse nous fournir la cascade décrivant un DS, nous devons construire un échantillon d'exemples positifs (i.e. composé d'images contiennant un sourire) - et un échantillon d'exemples négatifs (i.e. composé d'images qui ne contiennent pas un sourire) .

Le procédé exact que nous avons mis en œuvre pour aboutir aux premières versions d'échantillons SMILEs (*Smiling Multisource Incremental-Learning-Extensible sample*) est décrit dans l'article attaché en Annexe 3. Résumons-le en quelques mots:

Nous sommes parti de la base d'images "Labeled Faces in the Wild" (Huang et al. 2007) (LFW) qui contient 13080 images (c.f. figure 14) . Un petit logiciel était programmé pour permettre - et faciliter - le tri manuel de la LFW en deux groupes – des exemples positifs et négatifs. Ce logiciel nous permettait aussi, pour les exemples positifs, de marquer par une méthode facile de *click&drag&drop* la région d'intérêt (RdI) contenant la bouche souriante. Après quelques heures de travail assez exigeant[19] nous avons abouti à un échantillon positif contenant 3606 images et à un échantillon négatif contenant 9474 images. À partir de ces échantillons, *haartraining* nous a fourni la version 0.1 de notre détecteur de sourire (que nous avons appellé *smileD*). Ensuite nous avons mis à l'épreuve cette première version en l'appliquant aux nouvelles images dont nous étions sûrs qu'elles contenaient les sourires (i.e. les images faisant partie de la base Genki4K (Whitehill et al. 2007) ou/et les images que nous avions télechargées automatiquement du site flickr.com en cherchant le mot clé "smile"). Comme la première version de détecteur y a reconnu quelques sourires, c'est-à-dire des RdI contenant de sourires, nous pouvions étendre l'échantillon de base avec les nouvelles images, cette fois-ci labélisées sans qu'aucune intervention manuelle ne soit nécessaire.

Figure 14 : Quelques exemples "positifs" extraits de la base LFW

C'est justement pour cette raison que nous avons mis les termes *Incremental-Learning-Extensible* dans le titre de notre projet.

SMILEd

En fournissant les 5 versions différentes d'échantillons SMILEs au logiciel *haartraining* (c.f. Paragraphe E de l'article en Annexe 3 pour connaître les paramètres d'entraînement), nous avons obtenu 5 versions différentes du détecteur de sourire smileD.

SmileD est couplé avec un détecteur de visages. En d'autres termes, le logiciel essaye d'abord de détecter le visage et, s'il y reussit, il va chercher dans sa partie inférieure centrale[20] la bouche souriante. Ce couplage – que nous croyons raisonnable puisque nous n'avons pas encore vu de sourire qui ne soit imbriqué dans un visage – était pris en compte aussi pendant la période de

19 Exigeant d'un point de vue cognitif. En effet, il nous arriva souvent, après quelques heures consacrées à la démarcation de maintes et maintes RdI, de commencer à percevoir des sourires même là où il n'y en avait pas.

20 Le paragraphe 310 de (Da Vinci & Richter 1970) indique: "*The space between the parting of the lips [the mouth] and the base of the nose is one-seventh of the face...The space from the mouth to the bottom of the chin is the fourth part of the face and equal to the width of the mouth...The space from the parting of the lips to the top of the chin, that is where the chin ends and passes into the lower lip of the mouth, is the third of the distance from the parting of the lips to the bottom of the chin and is the twelfth part of the face. From the top to the bottom of the chin is the sixth part of the face and is the fifty fourth part of a man's height".*

construction d'échantillons d'apprentissage car nous avons mis seulement des images de visages sans sourire dans l'échantillon d'exemples négatifs. Cet échantillon n'était censé contenir que des images de fond (*background*).

De nos tentatives résultèrent cinq fichiers XML de 100 à 300 kilo-octets que chacun pourra soit ajuster à son propre gré, soit embarquer dans son logiciel, dès que nous les publierons en tant que package *open source*. Afin d'évaluer les deux versions (v0.1 et v0.5) du *smileD* dans les conditions quotidiennes nous conçûmes 2 expériences dans lesquelles Roboto nous rendit service.

Méthode

Expérience 1 - Participants

Quinze participants (11 hommes, 4 femmes) furent conviés à jouer au “jeu d'imitation”. Ils furent assis en face de Roboto, ils eurent pour consigne de “faire la même chose que le robot”. La distance entre le visage des participants et la caméra placée variait entre 50 et 100cm, selon les exigences dues au confort des participants.

Expérience 2 - Participant

Un seul participant (homme, 27 ans) joua au même jeu d'imitation que les participants de l'expérience 1. Il l'a effectué d'abord dans le mode “barbu” pour l'effectuer le lendemain en mode “rasé”. Pour chaque mode, il y avait deux séances - l'une avec une distance de 50cm, l'autre de 100cm. La luminosité d'environnement restait identique entre les séances.

Roboto

Pendant deux expériences, le mouvement de Roboto fut géré par un logiciel *immitation_game.c* écrit en C++ qui envoyait au robot les séquences codant quatre expressions: le sourire, la surprise, la tristesse et l'EF neutre (c.f. Figure 4 et Tableau 1). Chaque envoie de séquences, 23 au total pour chaque sujet, était suivi d'un intervalle temporel pendant lequel 42 images étaient enregistrées.

Afin de réduire les interférences entre les expressions successives, le sourire, la surprise et la tristesse étaient toujours suivis par l'EF neutre. Au contraire, l'expression neutre était toujours suivie par l'une des trois expressions affectives, leur ordre étant défini de manière aléatoire.

L'analyse des images

Les images furent divisées en deux classes: les positives supposées contenir un sourire puisque enregistrées après l'envoie de l'instruction “sourire”; et les négatives, enregistrées pendant l'intervalle suivant l'envoie de l'instruction ”surprise” ou “tristesse”.

Chaque image obtenue (i.e. 23x42=969 au total pour chaque sujet) fut analysée par le détecteur de visage *frontal_face* qui est fourni avec la bibliothèque OpenCV. Si un visage était détecté, les détecteurs de sourire smileD v0.1 et v0.5 sont mis à l'épreuve dans la région d'intêrêt définie par les trois cinquièmes centraux du tiers inférieur du visage. Quand smileD ne trouvait aucune région rectangulaire susceptible de contenir un sourire , le détecteur rendait la valeur 0. Au contraire, si une telle région est identifiée, la fonction cvHaarDetectObjects() qui est à la base de smileD rend le nombre de régions se recouvrant mutuellement, dont toutes sont susceptibles de contenir un sourire. Le nombre entier ainsi obtenu était appelé "l'intensité du sourire" par (Deniz et al. 2008) et nous aussi y référons par ce nom.

Les courbes ROC

Ayant ainsi une "intensité de sourire" pour chaque image où le sourire avait été detecté, nous pouvions utiliser cette quantité comme seuil de discrimination (*cutoff*) pour construire des courbes ROC (*Receiver Operating Characteristic*) . Les courbes ROC sont le moyen le plus commun pour représenter la performance totale d'un classifieur car elles représentent la performance du classifieur, définie par le nombre des VP et des FP par rapport aux VN respectivement FN, sous les conditions qui varient selon la valeur du seuil de discrimination. Comme la communauté autour de AA utilise souvent la mesure "aire sous la courbe" (ASC ou en anglais *AUC – area under curve*) pour comparer les classifieurs, nous avons calculé les valeurs d'ASC correspondantes grâce à la bibliothèque ROCR (Sing et al. 2005) du langage R (Team 2006).

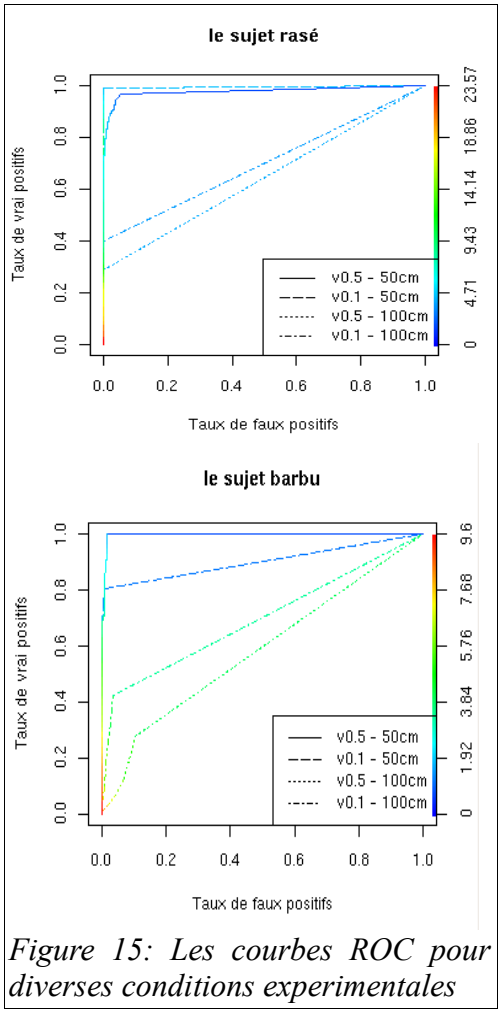


Figure 15: Les courbes ROC pour diverses conditions experimentales

Les 12 premières images de chaque séquence ne furent pas prises en compte lors de la construction des courbes ROC parce qu'il s'agissait de périodes transitoires entre deux EFs. Huit courbes ROC furent construites, une pour chaque combinaison des conditions expérimentales.

Résultats

Expérience 1

Les images enregistrées se sont avérées complétement inutilisables, car mal étiquettées en raison d'un bug dans le logiciel *immitation_game.c*.

Expérience 2

ASC pour la version 0.5 du détecteur smileD, quand mis à l'épreuve à 50 cm de distance, fut de 99.6% pour la séance pendant laquelle le sujet était barbu et de 97.75% quand il était rasé. Quant au *smileD* version 0.1, toujours à 50 cm de distance, la performance fut de 99.4% pour un sujet rasé mais seulement de 90% quand le sujet était barbu.

Les détecteurs se sont avérés moins performants quand le sujet était assis à un métre de la caméra de Roboto – plus exactement, 58.2% d'ASC pour le détecteur version 0.5 quand le sujet était barbu et 64.4% d'ASC quand le sujet était rasé. Pour smileD version 0.1, les résultats obtenus sont de 69.6% pour le mode barbu et de 70.1% pour le mode rasé.

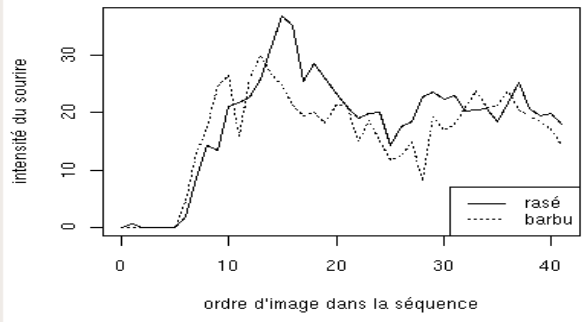


Figure 16 : L'évolution d'intensité du sourire dans le temps. Dans tous les deux cas, l'intensité du sourire culmine environ une seconde après l'expression de sourire par Roboto, puis retombe vers les sourires considérés par détecteur comme "moins marqués", puis remonte de nouveau...

La figure 16 montre l'évolution de la quantité "intensité de sourire" à travers la séquence vidéo enregistrée après l'envoi de l'instruction "sourire" au Roboto. Seules les données avec le facteur Distance de 50 cm furent prises en compte. Environ 7 images après l'envoie de l'instruction, nous constatons une augmentation brusque d'intensité de sourire jusqu'à un "peak" atteint entre la dixiéme et la vingtiéme image[21]. Notons que, lors de la construction de la figure 16, les valeurs d'intensité obtenues furent moyénées à travers les séances.

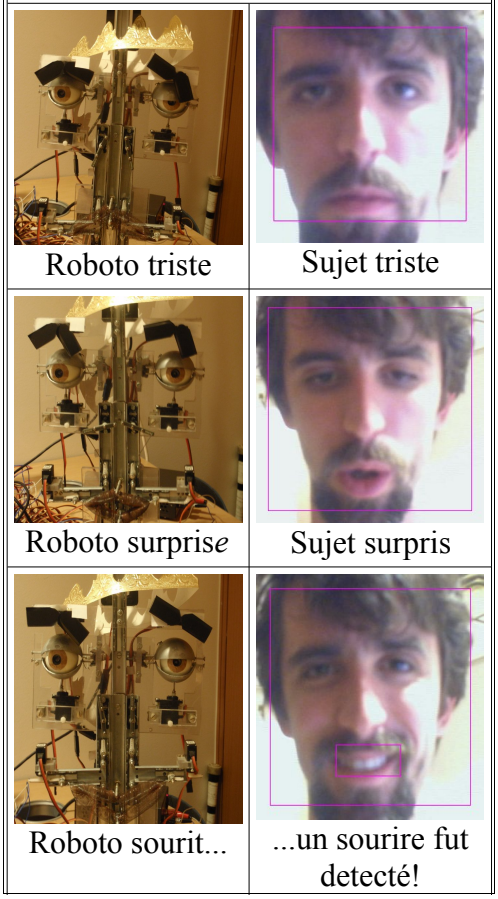
Discussion

Dans la communauté d'AA, la mesure ASC est souvent interprétée comme "la probabilité que le classifieur attribuer un score plus élevé à l'exemple positif qu'au négatif, tous les deux étant choisis de manière aléatoire" (Fawcett, 2006). Étant donné que dans le cadre de cette recherche le terme score est un synonyme pour l'intensité du sourire, nous constatons que nous avons réussi à construire un tel DS (*smileD v0.5*) qui attribuerait, dans plus de 99,6% des cas, une intensité du sourire plus élevée à n'importe quelle images enregistrée lors de l'intervalle temporel suivant l'envoi d'instruction "sourire" à Roboto plutôt qu'à celle enregistrée lors de l'intervalle temporel suivant l'envoi d'instruction "tristesse", "surprise" ou "neutre".

21 Étant donné que la vitesse d'enregistrement était environ 15 images per seconde, le pic d'intensité fut atteint à peu prés 1 second après l'envoi d'instruction au Robot. Malheureusement toutes les interprétations de figure 16 dans les termes d'un "temps de réaction" absolu doivent être rejetées comme imprécises puisque l'ordre d'image dans la séquence ne nous donne que des renseignements indirects sur le tampon temporelle d'image enregistrée. Ceci est du au fait que la vitesse d'enregistrement varie selon l'état d'ordinateur dans le moment d'expérience et on ne peut pas donc en inférer les coordonnées temporelles exactes. D'où l'importance d'ajouter le code qui rendra possible l'enregistrement des données temporelles – avec la précision en miliseconds - dans la prochaine version du *immitation_game.c*

Qui plus est, la similarité valeurs d'ASC pour les modes “rasé” et “barbu” indique que nous avons, en effet, construit un DS robuste contre certaine variabilité propre à l'objet à reconnaître. Cette proposition est d'ailleurs soutenue par les résultats de l'article en Annexe 3 qui montrent que la performance de smileD s'élève à plus de 90% quand on l'a confronté avec la base d'images JAFFE. Notons enfin que ni les images faisant partie de JAFFE, ni aucune des images du participant qui a du subit l'expérience “barbu/rasé” (c.f. Figure 17) ne faisaient partie de l'échantillon d'apprentissage.

Figure 17 : Le jeu d'imitation entre Roboto et le sujet barbu



Par contre, le fait que le DS présenté ci-dessus ne donne que des très faibles performances quand il est mis à l'épreuve contre les images prises d'une distance de 100 cm indique l'une des faiblesses des premières versions de *smileD*. Comme l'approche de la reconnaissance d'objets par le moyen des caractéristiques rectangulaires de Haar est supposée être invariable par rapport à la taille (et donc par rapport à la distance) de l'objet à reconnaître (Bradski & Kaehler 2008), les résultats obtenus nous indiquent que les versions de *smileD* construites jusqu'au ici sont loin d'être les solutions finales.

Cependant, nous avons des raisons de croire que le problème lié à la taille du sourire sera résolu dans de semaines à venir. Non seulement nous croyons, enfin, comprendre la théorie de Viola & Jones aussi bien que les subtilités du logiciel *haartraining*, mais aussi nous soupçonnons que le problème mentionné ci-dessus est dû au fait que nous avons utilisé des valeurs trop élevées de largeur=43 et hauteur=19 comme paramètres d'apprentissage pour les cinq premières versions de *smileD*.

Avant cela, nous croyons que le projet *SMILEsmileD*, contenant aussi bien le DS *smileD* qu'un échantillon d'apprentissage *SMILEs*, a certaines chances de réussir, si jamais il est remarqué par la communauté internationale qui l'affinera et l'enrichira, en accord avec la philosophie *open source*. Tels sont nos éspoirs (TSNE).

Les exploitations possibles d'un DS sont innombrables. Laissons de coté ses mises en application commerciales - comme les plugins pour l'interaction dans les réseaux sociaux; ou militaires – pour que le système puisse mieux reconnaître les individus nonconformes à la norme (Huxley 1969) et refusant de s'exprimer comme les vedettes de Holywood. Laissons de côté tout cela et réfléchissons sur deux utilisations dignes de l'appareil que nous avons tenté de construire.

La première utilisation est liée à la thérapie des troubles émotionnels et affectifs. Ceci est lié au phénomène que nous voyions se répeter maintes et maintes de fois depuis que Roboto a débarqué au Lutin : n'étant qu'un tas de ferraille, son sourire a toujours fait sourire ceux qui l'ont regardé. Nous croyons que la force de ce phénomène ne peut qu'augmenter, si jamais une véritable imitation – une véritable harmonisation temporelle entre l'homme et la machine – se met en place.

L'auto-catalyse de la bonne humeur portera ses fruits, un DS robuste étant le premier pas vers cet objectif.

La deuxième utilisation est liée au domaine d'intelligence artificielle (IA), plus précisément en IA développementale où l'utilisation d'un DS est rapporté par (Movellan et al. 2007)– ou dans le domaine de la pédagogie des machines. Étant donné que 1) le sourire est un moyen naturel grâce auquel un être humain, *un instituteur humain*, exprime son contentement; et étant donné que 2) les premiers instituteurs des machines sont et seront les êtres humains, un sourire nous paraît être le moyen le plus approprié de renforcement positif (Skinner 1976) du comportement des machines.

Le principe en est assez simple : l'algorithme donnera plus de poids aux représentations d'actions effectuées et situations perçues qui se succèdent de manière immédiate par un sourire..

Voilà deux utilisations où sourire joue le rôle principal.

Sourire est un don qui permet aux humains de devenir plus humains, un don qui leur permet de franchir leurs syndromes, leurs maladies, la haine, voire la mort même.

Et qui sait si, un jour, il ne le permettra même aux machines ?

TSNE.

Bibliographie

Akita, M., Marukawa, K. & Tanaka, S., 2010. Imaging apparatus and display control method.

Bradski, G. & Kaehler, A., 2008. *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media.

Darwin, C., 1872. The expression of the emotions in man and animals; with an introduction, afterword, and commentaries by Paul Ekman. *NY: Oxford University*.

Da Vinci, L. & Richter, J.P., 1970. *The notebooks of Leonardo da Vinci*, Dover Publications.

Dawson, G. et al., 1990. Affective exchanges between young autistic children and their mothers. *Journal of Abnormal Child Psychology*, 18(3), 335–345.

Deniz, O. et al., Smile Detection for User Interfaces. *Advances in Visual Computing*, 602–611.

Duchenne de Bologne, G.B., 1862. The mechanism of human facial expression. *Paris: Jules Renard*.

Ekman, P. & Friesen, W.V., 1982. Felt, false, and miserable smiles. *Journal of Nonverbal Behavior*, 6(4).

Ekman, P. & Friesen, W.V., 1977. Manual for the facial action coding system. *Consulting Psychologist*.

Fawcett, T., 2006. An introduction to ROC analysis. *Pattern recognition letters*, 27(8), 861–874.

Huang, G.B. et al., 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *University of Massachusetts, Amherst, Technical Report*, 57(2), 07–49.

Huxley, A., 1969. Brave New World. 1932. *New York: HarperPerennial*, 246.

Jack, R.E. et al., 2009. Cultural confusions show that facial expressions are not universal. *Current Biology*.

Movellan, J.R. et al., 2007. The RUBI project: a progress report. Dans *Proceedings of the ACM/IEEE international conference on Human-robot interaction*. p. 339.

Sing, T. et al., 2005. ROCR: visualizing classifier performance in R. *Bioinformatics*.

Skinner, B.F., 1976. Walden two revisited. *BF Skinner, Walden Two (reissued)*.

Strathearn, L. et al., 2008. What's in a smile? Maternal brain responses to infant facial cues. *Pediatrics*, 122(1).

Team, R.D.C., 2006. R: A language and environment for statistical computing. .

Viola, P. & Jones, M., Rapid Object Detection using a Boosted Cascade of Simple. Dans *Proc. IEEE CVPR 2001*.

Whitehill, J. et al., 2007. Developing a practical smile detector. *Submitted to PAMI*, 3, 5.

Annexe 1. Le fonctionnement d'algorithme AdaBoost

Les entrées: L'échantillon d'entraînement

$$\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$$

Distribution D liée au N exemples ayant d_0 d_i pour comme étiquettes

Le modèle faible d'apprentissage

L'entier T spécifient le nombre d'iterations d'algorithme

L'initialisation: Mettre $D_1(i)=1/N$ pour tout I

La computation: Faire le suivant pour n=1,2,...,T :

1. Appeler le modèle faible d'apprentissage en lui fournissant la distribution D_n
2. Obtenir comme réponse une hypothèse Hn : $X \rightarrow Y$
3. Calculer l'erreur d'hypothèse H_n : $\epsilon_n = \sum_{i: H_n(x_i) \neq d_i} D_n(i)$
4. Mettre $\beta_i = \epsilon_n / 1 - \epsilon_n$
5. Mettre à jour la distibution D_n :

$$\mathcal{D}_{n+1}(i) = \frac{\mathcal{D}_n(i)}{Z_n} \times \begin{cases} \beta_n & \text{if } \mathcal{F}_n(\mathbf{x}_i) = d_i \\ 1 & \text{otherwise} \end{cases}$$

ou Z_n est une constante de normalisation (choisie de telle manière que D_{n+1} est une distribution de probabilité)

La sortie: L'hypothèse finale – le classifier fort - est :

$$\mathcal{F}_n(\mathbf{x}) = \arg\max_{d \in \mathcal{D}} \sum_{n: \mathcal{F}_n(\mathbf{x}) = d} \log \frac{1}{\beta_n}$$

Annexe 2. La construction d'images de *chamfer* en OpenCV

L'algorithme (c++)pour calculer l'image de chamfer en utilisant les routines de la bibliothèque OpenCV

```
//devant d'abord (au debut contour_image=chamfer)
for (i=1;i<(chamfer->height)-1;i++) {
  for (j=1;j<(chamfer->width)-1;j++) {
    s=cvGet2D(chamfer,i,j);
    int a[4];
    a[0]=s.val[0];
    a[1]=cvGet2D(chamfer,i-1,j).val[0]-2;
    a[2]=cvGet2D(chamfer,i-1,j-1).val[0]-3;
    a[3]=cvGet2D(chamfer,i,j-1).val[0]-2;
    a[4]=cvGet2D(chamfer,i+1,j-1).val[0]-3;
    max=max4(a);
    s.val[0]=max;
    cvSet2D(chamfer,i,j,s);
  }
}
```

```
//et apres en arriere
for (i=(chamfer->height)-2;i>0;i--) {
  for (j=(chamfer->width)-2;j>0;j--) {
    s=cvGet2D(chamfer,i,j);
    int a[4];
    a[0]=s.val[0];
    a[1]=cvGet2D(chamfer,i+1,j).val[0]-2;
    a[2]=cvGet2D(chamfer,i+1,j+1).val[0]-3;
    a[3]=cvGet2D(chamfer,i,j+1).val[0]-2;
    a[4]=cvGet2D(chamfer,i-1,j+1).val[0]-3;
    max=max4(a);
    s.val[0]=max;
    cvSet2D(chamfer,i,j,s);
  }
}
```

Annexe 3.

Semi-supervised haartraining of a fast&frugal open source zygomatic smile detector

A gift to OpenCV community

Daniel Devatman Hromada
Lutin Userlab
Ecole Pratique des Hautes Etudes

prof. Charles Tijus
Cognition Humaine et Artificielle (ChART)
Université Paris 8

***Abstract*—Five different versions OpenCV-positive XML haarcascades of zygomatic smile-detectors as well as five SMILEsamples from which these detectors were derived had been trained and are presented hereby as a new open source package. Samples have been extended in an incremental learning fashion, exploiting previously trained detector in order to add and label new elements of positive example set. After coupling with already known face detector, overall AUC performance ranges between 77%-90.5% when tested on JAFFE dataset and <1ms per frame speed is achieved when tested on webcam videos.**

Keywords-zygomatic smile detector; cascade of haar feature classifiers; computer vision; semi-supervised machine learning

I. Introduction

Great amount of work is being done in the domain of facial expression (FE) recognition. Of particular interest is a FE being at the very base of mother-baby interaction [1], a FE interpreted unequivocally in all human cultures [2] - smile. Maybe because of these reasons, maybe because of some others, smile detection is already of certain interest for computer vision (CV) community – be it for camera's smile shutter [3] or in order to study robot2children interaction [4].

Nonetheless, a publicly available i.e. open source, smile detector is missing. This is somewhat stunning, especially given the fact that "smile" can be conceived as a "blocky" object [5] upon which a machine learning technique based on training of cascades of boosted haar-feature classifiers [6] can be applied, and that the tools for performing such a training are already publicly available as part of an OpenCV[5] project. Verily, with exceptions of detectors described in [7][8] which have not been publicly released, we did not find any reference to haarcascade-based smile detector in the literature. We aim to address this issue by making publicly available the initial results of our attempts to construct sufficiently descriptive SMILing Multisource Incremental-Learning Extensible Sample (SMILEs) and five smile detectors (smileD) generated from this sample.

From more general perspective, our aim was to study whether one can use already generated classifiers in order to facilitate such a semi-supervised extension of initial sample that a more accurate classifier can be subsequently trained.

A. *SMILE sample (SMILEs)*

The aim of SMILEs project is to facilitate and accelerate the construction of smile detectors to anyone willing to do so. Since it is the OpenCV library which dominates the computer vision community, SMILEs package is adapted upon the needs of OpenCV in a sens that it contains 1) negative examples directory 2) positive examples directory 3) negatives.idx - list of files in negative examples directory 4) positives.idx - list of files in positives with associated information containing the coordinates of region of interest (ROI), i.e. the coordinates of the region within which smile can be located.

SMILEs is considered "Multisource" because it originates as an amalgam of already existing datasets like LFW and Genki both of which are, themselves, collections of images downloaded from the Internet. Images from POFA [9] of Cohn-Kanade [10] datasets were not included into SMILEs since restricted access to these datasets is in contradiction with an open source approach[1] of SMILEs project.

B. *Smile Detector (smileD)*

SMILEs are "Incremental-Learning Extensible" in a sense that they allow us to train new versions of smile detectors which are subsequently applied upon new image datasets in order to facilitate (or even fully automatize) the labeling of new images, and hence extending an original SMILEs with new images. Simply stated, SMILEs allow us train smileD which helps us to extend SMILEs etc.

Since training of haar cascades is an exhaustive threshrold-finding process demanding not negligible amount of time and computational resources, 5 pregenerated OpenCV-compatible XML smileD haarcascades were trained by opencv-haartraining application and are included with SMILEs in our OpenSource SMILEsmileD package, so that

[1] Both SMILEs & SMILEd cascades are publicly available from http://github.com/hromi/SMILEsmileD as a GPL-licensed package. C++ source codes of select&crop application for easy manual sample creation and of a face-coupled video stream smile detector are included as well.

anybody interested could implement our smile detector in copy&use fashion.

II. Method

C. Initial Training Datasets

SMILEs project in its current state unites 3 image sets :

- cropped Labeled Faces in the Wild (LFW) dataset - LFW dataset [11] contains more than 13000 images of faces collected from the web; its cropped version contains only 25x25pixel regions detected by OpenCV's frontal face detector. No information about the presence/absence of a sm.ile within the image is given
- Genki4K dataset – Genki4K is a publicly available part of UCSD's Genki project [12] containing 4000 images downloaded from Internet. A text file indicating the presence/absence of the smile in a given image is included.
- Ad hoc Flickr dataset – We have used the search keyword "smile" in order to download more than 4200 additional pictures from image-sharing website flickr.com. More than 2600 of them contained at least one smiling face.

D. Construction of SMILEs datasets

We have created five different version of SMILEs. All these versions exploit the same negative sample set of LFW's nonsmiling images. All manual labeling focalised solely on zygomatic smile (ZS) region[2]:

- **Version 0.1** is based solely upon an LFW dataset. All pictures were manually labeled by our ad hoc region selection & cropping application and divided into samples of positive (3606 images) and negative (9474 images) examples.
- **Version 0.2** added 2666 manually labeled images downloaded from flickr.com to positive examples contained already in 0.1. Labeling & region selection was realised by same application as in case of 0.1.
- **Version 0.3** also extended the positive&negative example samples of version 0.1 with images from flickr. This time, however, the flickr-originated images weren't labeled manually, but the smile-containing regions of interest were determined automatically, by applying smileD of version 0.1 upon the set of downloaded images. 1372 ROIs (1 ROI for 1image) were identified&labeled in this way.
- **Version 04** is analogous to version 0.3 in that sense that it is essentially a version 0.1 sample to which automatically labeled positive examples were added. Differently from version 0.3, Genki4K and not flickr was exploited as a source of additional data. Simply stated, positive examples, 624 of them in total, from Genki4K labeled as smile-containing by its authors were added to initial LFW-based sample.
- **Version 05** unites the versions 0.3 and 0.4, i.e. both Genki4K & flickr-originated images which were automatically labeled by smileD v0.1 were added to LFW samples.

TABLE I. Basic Components of initial versions of smiles&smiled project

Version	Positive examples					Neg. ex.
	LFW manual	*Flickr manual*	*Flickr auto*	*Genki auto*	*Total*	
0.1	3606	0	0	0	3606	9474
0.2	3606	2666	0	0	6262	9474
0.3	3606	0	1372	0	4978	9474
0.4	3606	0	0	624	4230	9474
0.5	3606	0	1372	624	6572	9474

[2] ZS region was defined only vaguely as a rectangular ROI in whose center are smiling lips – in preference with uncovered teeth. Whole ROI is bordered by smile&nasolabial wrinkles.

E. SMILEs -> smileD Training

Identical haarcascade training parameters [width=43, height=19, number of stages=16, stage hit rate=0.995, stage false alarm rate=0.5, week classifier decision tree depth=1(i.e. Stump), weight trimming rate=0.95] were applied for training of all five smileD versions, one smileD corresponding to one SMILEs, both referenced by same version number.

F. smileD evaluation

Training phase of every new version of smileD was followed by measuring its performance upon a Japanese Female Facial Expression (JAFFE) dataset in order to evaluate the performance of different versions of smileD classifiers when applied upon a sample having different luminosity conditions than that any imageset included in train sample

Detectors were face-detector-coupled during testing, i.e. smile detection was performed iff a face was detected in a tested image, and only in the ROI defined by well-known geometric ratios [13]

Receiver operating characteristic (ROC) curves were plotted and AUC ("area under ROC curve") were calculated as performance measures by means of ROCR library [14]. "Smile intensity" [7], i.e. the number of overlapping neighboring hit regions[3], was used as a cutoff parameter.

III. Results

FIGURE I. SmileD roc curves

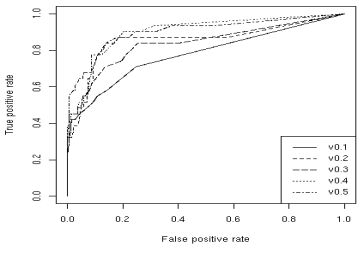


TABLE II. Roc's "area under curve" performance of different versions of smileD detector

Version	AUC
0.1	77.94%
0.2	85.49%
0.3	83.93%
0.4	90.21%
0.5	90.51%

[3] Can be obtained from undocumented *neighbors* attribute of *cvAvgComp* sequence referenced by *cvHaarDetectObjects*

Discussion

Detectors we present hereby exploit the top-bottom approach, i.e. they are face-coupled. Knowing that *there can be no smile without the face within which it is nested*, we firstly detect the face by an OpenCV face detection solution and then smileD is applied only in very limited ROI of face's bottom third. Consequences of our decision to create face-coupled smile detector are twofold: 1) since by definition we search for smile only within the face, we have used only nonsmiling faces as negative examples (i.e. background images) 2) smile detection itself is very fast, once the position of face is specified. When applied upon the webcam-originated (320x240 resolution) video streams, the time needed for smile detection never exceeded 1ms per frame on a Mobile Intel(R) Pentium(R) 4 CPU (1.8GHz), suggesting that our detector could be potentially embedded even into mobile devices disposing with lesser computational resources.

SmileD's speed can somehow neutralize its smaller accuracy handicap which it has in comparison with results reported in [8]. In its current state, our approach suffer from somewhat high false alarm rates, but our research indicates that in real life condition, these can be in great measure reduced by taking into account the dynamic sequence of subsequent frames since the probability of the same false alarm occuring within all the frames of the sequence is proportional to the product of probabilities of occurrence of that false alarm for every frame of the sequence taken individually. High speed is therefore of utmost importance and analysis of sequences of frames can substantially reduce the number of false positives.

Tuning of training parameters and the extension of negative example do remain as other possibilities how to augment the accuracy of our project. Tab.2 indicates that accuracy of such semi-supervised classifiers like smileD gets saturated at certain limit which can possibly be surmounted only by extension of negative sample set. In case of smile detection, we suggest that extension of negative example sample with more images containing "upper lip raiser" action unit (AU 10) – teeth-uncovering[4] but associated with disgust rather than smile – could yield some significant increases in accuracy, as reported by [9]. Since such an extension is relatively easy and not much time-consuming, given that such AU10-containing images are given and marked as negative examples, it may be the subject of future research.

In this study, however, we left the negative example unchanged in order to study the effectivity of "Incremental Learning" approach during which an old detector is used to facilitate the extension of a positive example sample thanks to which a new detector is obtained. Since semi-supervised smileD versions v0.4 and v0.5 have outperformed v0.2 for which manual labeling was implemented, while the latter one performed only slightly better than v0.3 which exploited an identic flickr-originated imagebase than v0.2, it is not unreasonable to think that such semi-supervised incremental training approach can be a feasible solution for training haarcascade detectors. If that would be the case, it could possibly be stated that the machine started, in certain sense, to ground [15] its *own of smile*.

[4] From anatomic point of view, disgust-expressing AU10 is associated with *Levator Labii Superioris* muscle while smile associates with *Zygomaticus Major* muscle (AU12).

AcknowLedgment

We would like to thank the third section of EPHE, University Paris 8 and CROUS de Paris for their kind support.

References

[1] L. Strathearn, J. Li, P. Fonagy, et P.R. Montague, "What's in a smile? Maternal brain responses to infant facial cues," *Pediatrics*, vol. 122, 2008, p. 40.

[2] C. Darwin, P. Ekman, et P. Prodger, *The expression of the emotions in man and animals*, Oxford University Press, USA, 2002.

[3] M. Akita, K. Marukawa, et S. Tanaka, "Imaging apparatus and display control method," 2010.

[4] J.R. Movellan, F. Tanaka, I.R. Fasel, C. Taylor, P. Ruvolo, et M. Eckhardt, "The RUBI project: a progress report," *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, 2007, p. 339.

[5] G. Bradski et A. Kaehler, *Learning OpenCV*, O'Reilly Media, Inc., 2008.

[6] P. Viola et M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple," *Proc. IEEE CVPR 2001*.

[7] O. Deniz, M. Castrillon, J. Lorenzo, L. Anton, et G. Bueno, "Smile Detection for User Interfaces," *Advances in Visual Computing*, p. 602–611.

[8] J. Whitehill, M. Bartlett, G. Littlewort, I. Fasel, et J. Movellan, "Developing a practical smile detector," *Submitted to PAMI*, vol. 3, 2007, p. 5.

[9] P. Ekman et W.V. Friesen, *Pictures of facial affect*, Palo Alto, CA: Consulting Psychologists Press, 1976.

[10] T. Kanade, Y. Tian, et J.F. Cohn, "Comprehensive database for facial expression analysis," *fg*, 2000, p. 46.

[11] G.B. Huang, M. Ramesh, T. Berg, et E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition" *University of Massachusetts, Amherst, Technical Report*, vol. 57, 2007, p. 07–49.

[12] J. Whitehill, G. Littlewort, I. Fasel, M. Bartlett, et J. Movellan, "Toward Practical Smile Detection," *IEEE transactions on pattern analysis and machine intelligence*, 2009, p. 2106–2111.

[13] L. Da Vinci et J.P. Richter, *The notebooks of Leonardo da Vinci*, Dover Publications, 1970.

[14] T. Sing, O. Sander, N. Beerenwinkel, et T. Lengauer, "ROCR: visualizing classifier performance in R," *Bioinformatics*, 2005.

[15] S. Harnad, "The symbol grounding problem," *Physica d*, vol. 42, 1990, p. 335–346.

Épiprologue[22]

Monsieur Charles Tijus
LUTIN - UMS CNRS 2809
Cité des Sciences et de l'Industrie

Lors mon stage en M2 des études CNA SVT à E.P.H.E. je voudrais **apprendre à robots de sourire**. Pour le faire de manière réussite, il faut d'abord répondre à deux questions: Comment? et Quand?

Pour répondre à la question « Comment? » je devrai: étudier la théorie des expressions émotionnelles, je devrai comprendre que se passe-t-il sur le visage quand on sourit: « quels muscles sont détendus? quels sont contractés? » etc . Bref, j'étudierai la théorie des émotions et leurs expressions faciales.

Ensuite, pour approcher cette théorie à la réalité des robots, j'étudierai le manuel et le fonctionnement et le « instruction set » du visage robotique « Roboto ». Je créerai un petit script en langue de programmation PERL (output du première semestre???) grâce auquel je serai capable d'envoyer les commandes comme <sourire>,<colere> à ce visage.

Quant à la question « Quand la[23] robote va-t-elle sourire? » je réponds tout de suite: Elle va sourire quand l'être humaine qu'elle voit, sourit à elle. Ainsi, la robote va imiter l'être humaine en face d'elle, *tel un bébé* qui fait la même chose dès ces premières moments dans ce monde. Même si la question de base est répondu, l'aspect technique de cette imitation pose des problèmes auquel je voudrai essayer à répondre pendant mon stage.

Afin pouvoir mimer, la robote devra d'abord reconnaître quelle émotion à mimer. Il faudra donc trouver le moyen de reconnaître l'émotion à partir des données contenus dans la photo prise par les cameras dans les yeux de la robote. Car dans notre monde c'est certain que quelqu'un s'est déjà posé la question et l'a répondu au moins partiellement, j'ai d'abord étudierai « l'état de l'art de reconnaissance automatique des émotions faciales » pour ensuite choisir la méthode approprié pour y aboutir (les candidats du départ sont pour cette méthode sont: les réseaux nerveux artificiels, SVM (support vecteur machine), la bibliothèque openCV ou le logiciel faceAPI; ou bien en système hybride de ces solutions).

Mon objectif sera surtout un logiciel open source ou une bibliothèque PERL (output du deuxième semestre???) qui permettra aux autres chercheurs d'interagir avec les machines par le moyen des expressions faciales des émotions de manière plus facile et efficace. Ensuite, le visage robotique Roboto, capable de mimer au moins 2-3 émotions de base, pourrait être utilisé non seulement dans le cadre de plusieurs expériences de recherche, y compris l'intelligence émotionnelle des enfants autistes mais aussi comme un pont vers les êtres artificiels dont quelques s'approchent de plus en plus d'essence même de l'humanité.

Paris **18/06/2009**
Daniel D. Hromada

22 Nous laissons cette “proposition de stage” dans sa version originalle, i.e. avec toutes ses fautes d'ortographe.

23 Ce n'est qu'ici où nous disons en haute voix ce que nous voulions dire depuis le debut de nos travaux: le mot *robot* viens d'ouvrage R.U.R d'auteur Karel Čapek et est dérivé du mot commun aux plusieurs langes slaves, celui de [*robota*] signifiant “le travail” ou mieux même, “la corvèe”. Ce mot est du genre féminine (paradigme de déclinaision: /žena/, i.e. femme). C'est peut être pout cette raison que nous considerions, tout au long de notre stage, Roboto comme un être d'essence féminine plutôt que masculine. Ceci dit, il n'y a rien à ajouter, sauf ...