Master Informatique

« Suivi de visage et détection de saillances »

Rapport
en vue de la validation de
l'UE Initiation à la recherche
2018 - 2019

Étudiants:
Alaa Eddine BENKARRAD
Abdelaziz CHERIFI
Walid HAFIANE

Encadrants :
Monsieur Amine BOUMAZA
Monsieur Alain DUTECH
Monsieur Yann BONIFACE

Soutenu le 20/05/2019

Décharge de responsabilités

L'Université de Lorraine n'entend donner ni approbation ni improbation aux opinions émises dans ce rapport, ces opinions devant être considérées comme propres à leur auteur

Remerciements

En tout premier lieu, nous souhaitons remercier l'ensemble des enseignants de l'université de Lorraine ayant contribué à notre formation.

Nous sommes également très reconnaissants envers nos encadrants, Monsieur Amine Boumaza, Monsieur Alain Dutech et Monsieur Yann Boniface, pour le temps et la confiance qu'ils nous ont accordés ainsi que pour leurs précieux conseils, leur constante bonne humeur, et leurs moments très agréables passés avec eux.

Bien sûr, nous n'oublions pas de remercier nos familles et nos amis respectifs pour leur présence et leur soutien inconditionnel.

Table des matières

Table des Figures i
Table des Symboles ii
Introduction générale 1
Chapitre 1: Présentation du projet 2
Introduction . . . 3
1.1 Projet PsyPhINe . . . 3
1.1.1 Idée de base du projet . . . 3
1.1.2 Présentation du projet . . . 3
1.1.3 Projet similaire - Pinokio . . . 4
1.2 Étude de la lampe . . . 5
1.2.1 Description de la lampe . . . 5
1.2.2 Travaux réalisés . . . 6
1.3 Problématique . . . 7
Conclusion . . . 7
Chapitre 2: État de l'art 8
Introduction . . . 9
2.1 Méthodes de détection de visage . . . 9
2.1.1 Détection avec la méthode Haar cascade . . . 9
2.1.2 Détection avec les réseaux de neurones profonds (DNN) . . . 10
2.1.3 Détection avec la l'Histogrammes de gradients orientées (HOG) . 10
2.1.4 Détection avec les réseaux de neurones convolutionnel (CNN) . . . 12
2.2 Methode d'extraction des points de saillances . . . 13
2.3 Clustering . . . 15
2.3.1 K-means . . . 15
2.3.2 Carte auto-organisatrice (SOM) . . . 16
Conclusion . . . 18
Chapitre 3: Analyse et réalisation 19
Introduction . . . 20
3.1 Appoche globale . . . 20
3.2 Phase de préparation . . . 21
3.2.1 Prétraitement . . . 21
3.2.2 Détection du visage . . . 21
3.2.3 Éxtraction des points de saillances . . . 22
3.3 Phase d'extraction des VCC . . . 22
3.3.1 Éxtraction et encodagedes caractéristique . . . 22

3.3.2 Vecteurs caractéristiques des changements d'expressions 23
3.4 Phase de décision 24
3.4.1 K-Means 24
3.4.2 Carte auto-organisatrice dynamique 25
Conclusion 26
Conclusion générale **27**
Bibliographie **28**

Table des figures

1.1 Expreriences du projet Psyphine [1] 4
1.2 Pinokio la lampe de bureau rebotisée 4
1.3 La lampe de Psyphine [1] 5
1.4 Les moteurs de la lampe[1] 6

2.1 Caractéristiques de Haar 9
2.2 construction du gradient orienté 11
2.3 Processus de la construction de l'histogramme des gradients 12
2.4 visages détectés par le détecteur HOG et CNN 13
2.5 L'ensemble des 68 points détectés par Dlib pré-entraîné 14
2.6 Clustering 15
2.7 K-means clustering 16
2.8 Réduction de la dimensionnalité dans la SOM 17
2.9 Illustration de la décision de base à prendre lors de la croissance 18

3.1 Structure globale du systèùme 20
3.2 Prétraitement 21
3.3 Détection de visage 21
3.4 Éxtraction des points de saillances 22
3.5 Éxtraction des caractéristiques 23
3.6 Calcul des vecteurs caractèristiques des changements 24
3.7 Variante de l'algorithme k-moyennes 25
3.8 Structure de la DSOM utilisée 26

Liste des symboles

BMU Best Matching Unit

DNN Deep Neural Network

ERT Ensemble of Regression Trees

HOG Histogram of Oriented Gradients

MMOD Max Margin Object Detection

OpenCV Open Computer Vision

PsyPhINe Psychologie, Philosophie, Informatique et Neuroscience

RNA réseau de neurones artificiels

SVM Support Vector Machine

VCC Vecteurs Caracteristiques des Changements

Introduction générale

L'expansion rapide de l'industrie de l'intelligence artificielle et de la robotique est un facteur important qui influence et transforme divers aspects de notre vie quotidienne. Aujourd'hui, les machines intelligentes sont omniprésentes et elles sont devenues indispensables. Nous sommes de plus en plus souvent en relation avec des robots et des machines, que ce soit à des fins pratiques (thérapeutiques, professionnelles, scientifiques, quotidien ménager) ou ludiques. Par conséquent, ces relations, entre l'homme et la machine, entrainent de nombreuses questions et plus particulièrement sur l'**attribution d'intentions**, d'**intelligence** voire de **conscience** à un objet robotisé non humanoïde. Autrement dit, l'aspect humanoïde de la machine est-il nécessaire pour que nous soyons enclins à lui prêter des états mentaux ?. Afin d'étudier ces interactions et dans le but de répondre à ses questions liées au sujet, le projet Psyphine a été mis en place.

Le projet Psyphine regroupe plusieurs disciplines, à savoir, la psychologie, la philosophie, l'informatique et la neuroscience. Il s'interroge sur les **interactions** homme-robot et cherche à répondre à ces interrogations à travers plusieurs expériences. Pour ce faire, il utilise un **prototype robotisé** qui se présente sous la forme d'une lampe « La lampe Psyphine ». Cette dernière est un modèle unique qui a été construit et développé par le groupe.

Puisque le but du projet est d'étudier les **réactions** et les **comportements** des gens à travers des expériences dans laquelle les personnes seront en interaction non verbale avec la lampe rebotisé, notre objectif sera d'affecter à la lampe, la **capacité d'interagir** (de se comporter) conformément au **changement d'expressions du visage** de la personne dans l'expérience. La question que nous essayons d'y répondre est comment concevons-nous un système qui permet à la lampe d'exécuter des comportements suivant les différentes réactions des gens perçues par le biais de sa webcam ?.

Pour répondre à cet objectif, nous avons décidés d'organiser notre rapport en trois chapitres, tels que :

- Le premier chapitre sera une présentation du projet, de prototype utilisé et des différents travaux réalisés. A la fin de ce chapitre nous allons détaillés les différents aspects de notre problématique.
- Le seconde chapitre regroupe plusieurs notions, méthodes et techniques existantes dans l'état de l'art et spécifiquement celles que nous utiliserons dans notre méthode.
- Nous abordons, dans le troisième chapitre, les détails de notre solution dans laquelle nous expliquerons nos différents choix conceptuels et théoriques.

Chapitre 1

Présentation du projet

« ...la robotique et d'autres combinaisons rendront le monde assez fantastique comparé à aujourd'hui. ». B. Gates.

Sommaire

Introduction **3**
1.1 Projet PsyPhINe **3**
1.1.1 Idée de base du projet 3
1.1.2 Présentation du projet 3
1.1.3 Projet similaire - Pinokio 4
1.2 Étude de la lampe **5**
1.2.1 Description de la lampe 5
1.2.2 Travaux réalisés 6
1.3 Problématique **7**
Conclusion **7**

Introduction

Premièrement et avant d'expliquer notre méthode, il nécessaire d'introduire le contexte du projet et les moyens logiciels et matériels existants. Au cours du chapitre, nous présentons le projet Psyphine et d'autres projets similaires. Nous décrivons la structure physique de la lampe robotisée, ses caractéristiques, ainsi que les travaux réalisés dans le cadre du même projet. Nous clôturons ce chapitre par l'établissement de notre problématique sur laquelle nous construisons notre solution.

1.1 Projet PsyPhINe

1.1.1 Idée de base du projet

L'idée fondatrice du projet Psyphine est partie de l'interaction entre l'humaine et le robot de telle sorte que le robot peut se comporter de la même façon que l'humain en laissant le robot apprendre à travers son interaction avec l'humain et son environnement. Le but donc est que le robot tente de percevoir les intentions de l'être-humain avec lequel il interagit à partir d'indices multiples et d'expressions variées. Ainsi, cette perception s'étend aux interactions non verbales c'est-à-dire aux interactions non seulement avec des êtres humains capable de parler mais aussi avec des choses non humaines (animaux, objets). Pour cela il va falloir attribuer des comportements d'intentions, une certaine forme de volonté, et même des émotions dans certains cas à ce robot qui est programmé par l'homme. La diversité des différentes tâches du projet Psyphine a met en relation de nombreux chercheurs des différentes disciplines, tels que des psychologues, philosophes, roboticiens, cogniticiens, sociologues, et neuroscientifiques d'où le nom « PsyPhINe » qui est l'abréviation de « Psychologie, Philosophie, Informatique et neuroscience » [10].

1.1.2 Présentation du projet

Le projet PsyPhINe est lancé officiellement le premier avril 2016 par des chercheurs de l'université de lorraine, il vise à confronter et articuler les apports de différentes disciplines à la question de l'attribution d'intentionnalité, d'intelligence, de cognition voire d'émotions, à des entités naturelles ou des dispositifs artificiels. Il s'agit d'appréhender les questions naturellement posées par l'interaction homme/robot, à savoir celles liées à l'interprétation du comportement du robot jusqu'à la confiance qui peut lui être ou non accordée. Le projet vise notamment à explorer la gradation des attributions d'intelligence ou d'intentionnalité, quand on passe par exemple d'une mouche à un chat, en faisant l'hypothèse que l'intersubjectivité ainsi que notre tendance naturelle à l'anthropomorphisme jouent des rôles centraux : on projette dans l'autre énormément de notre propre cognition. La perspective générale du groupe de recherche pluridisciplinaire est d'aboutir à la définition d'un test de Turing non verbal qui permette d'appréhender l'intelligence artificielle en évitant certains écueils de la formulation d'origine dudit test. Dans le cadre du présent projet, le groupe PsyPhINe conduira des expérimentations à large échelle à l'aide d'un dispositif d'interaction basé sur un prototype de lampe robotisée. La mise en place des protocoles expérimentaux, des questionnaires et des analyses de vidéos, doublées

d'analyses « profanes », constituent le support et le lieu des confrontations et réflexions interdisciplinaires sur la cognition. Les résultats attendus en sont principalement une clarification d'un champ conceptuel riche et complexe, l'ébauche d'un langage commun et la production d'un référentiel des comportements donnant lieu à des interprétations intentionnelles. Comme présente la figure 1.1 suivante, la valorisation des travaux sera assurée par l'organisation d'ateliers interdisciplinaires et par l'édition d'un ouvrage collectif [10].

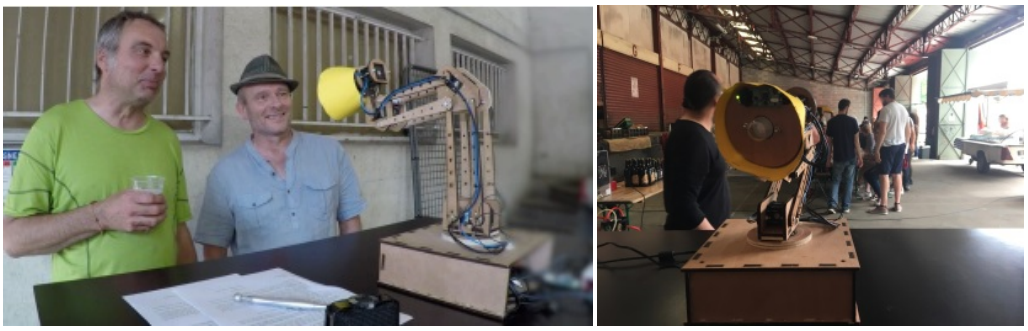


FIGURE 1.1 – Expreriences du projet Psyphine [1]

1.1.3 Projet similaire - Pinokio

Le nom Pinokio est venu de la lampe Luxo Jr, l'adorable petite lampe qui apparait dans le logo Disney Pixar, illustre comment les animateurs peuvent donner vie à des objets inanimés banals. Le trio de l'Université Victoria de Wellington, en Nouvelle-Zélande se sont inspiré de la lampe Luxo pour fabriquer leurs lampe robotisée grâce à une combinaison de robotique facilement accessible et de technologie de fabrication automatisée, combinée à des logiciels libres. Le trio qui comporte un programmeur Shanshan Zhou, un ingénieur mécanique Adam Ben-Dor qui travaillait sur les détails mécaniques de la lampe et un designer Joss Dogget qui s'est chargé du design et tout l'esthétique de la forme de la lampe, ce trio a donc décidé en 2012 d'embellir une lampe de bureau avec un peu de personnalité appelée Pinokio (présente dans la figure 1.2).

FIGURE 1.2 – Pinokio la lampe de bureau rebotisée

La lampe Pinokio est composée dans sa structure générale de son corps d'une tête,

d'un bras pliant et capable d'étirer et de rétrécir, une base sur laquelle le bras tourne et 6 servo-moteurs. Elle est également dite une lampe active (qui se déplace), ses actions sont principalement pilotées par Arduino et le logiciel de traitement d'image OpenCV, qui recherche le visage dans les images à partir de sa webcam. Lorsqu'il trouve un visage, il tente de le suivre comme s'il essayait de maintenir un contact visuel.

1.2 Étude de la lampe

Afin de répondre aux questions liées aux interactions homme/machine et l'attribution de conscience dans le cadre du projet, le groupe Psyphine a construit et développé un prototype robotisé qui se présente sous la forme d'une lampe dite « La lampe de Psyphine ». La structure et le modèle de fonction de cette lampe est inspiré de la lampe Pinokio décrite précédement.

1.2.1 Description de la lampe

Lampe de Psyphine, comme la montre la figure 1.3, est construite de contreplaqué léger dont les pièces ont été découpées au laser. Ces derniers sont assemblées autour de cinq moteurs. L'abat-jour a été découpé dans un carton jaune et contient une ampoule centrale et une petite caméra située juste au dessus de l'ampoule [1].

FIGURE 1.3 – La lampe de Psyphine [1]

1.2.1.1 Moteurs

La lampe est équipée de cinq moteurs de la marque Robotis Dynamixel en deux modèles, AX-12 et AX-18. Ces moteurs permettent cinq articulations (illustré par la figure 1.4) :

— **Moteur 1 :** la base sur un axe vertical
— **Moteur 2 :** le premier bras pour avancer ou éloigner la lampe

— **Moteur 3 :** le deuxième bras pour monter ou descendre l'abat-jour
— **Moteur 4 :** pour incliner l'abat-jour vers le bas ou vers le haut
— **Moteur 5 :** pour tourner l'abat-jour à droite ou à gauche

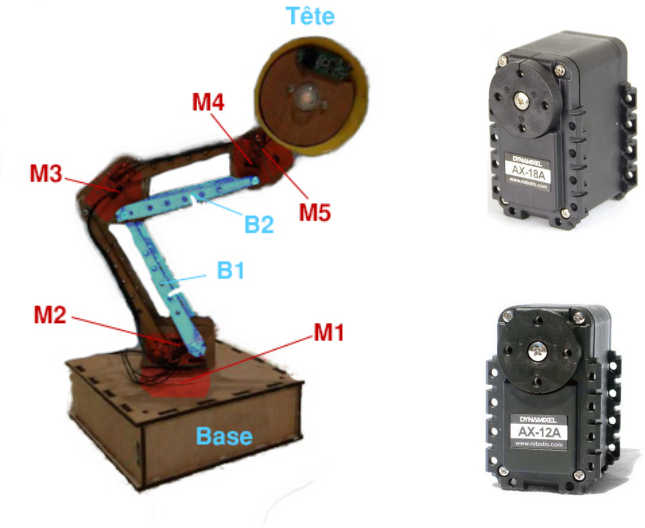


FIGURE 1.4 – Les moteurs de la lampe[1]

1.2.1.2 Système

La lampe comporte un système qui permet de faire fonctionner les moteurs. Cependant, la totalité des traitements couteux sont exécutés sur un ordinateur auquel les moteurs sont raccordés sur un port USB ou en wifi par l'intermédiaire d'un Raspberry Pi. Cet ordinateur s'occupe de commander la lampe rebotisé dont chaque mouvement de cette dernière consiste en une séquence de positions pour chacun des moteurs.

1.2.2 Travaux réalisés

Dans le cadre du projet, une panoplie de travaux ont été réalisés par des étudiants. Nous en citons :

Le travail [5] réalisé par D. SAMY et al. sur plusieurs axes. En premier, Ils ont mis en place un système pour contrôler les moteurs de la lampe à distance. Ensuite, ils ont réalisé un suivi de visage dont la détection est basée sur la méthode Viola et Jones implémenté dans la bibliothèque OpenCV. Finalement, leur travail à révélé plusieurs problèmes comme la fluidité de mouvement de suivi, qui a été réglé en modifiant le temps d'exécution, ainsi que le problème persistant de réchauffement des moteurs.

S. BARBILLON et al. ont construit, dans [2], une nouvelle architecture à l'aide de la bibliothèque Poppy pour faciliter la définition des nouveaux mouvements (interface de haut niveaux). Ils ont adapté la même méthode que celle du travail précédent pour le suivi de visage. Ainsi, ils ont consacré une grande partie de leur travail pour la cinématique inversé pour les mouvements de la lampe.

Un autre travail de A. Eva et G. Augustin [4] qui s'est concentré sur le suivi du visage et la détection des changements d'expressions. La méthode de Viola et Jones a été appliqué pour la détection du visage avec des amélioration au niveau de la détection de la bouche. Cette phase a été suivi par l'application de la régression linéaire pour détecter l'ouverture de la bouche.

R. BUISINE et O. RIOU ont exploré, dans leur rapport [11], un axe de travail différents des autres travaux. Il s'agit de la classification des émotions. Ils ont commencé par l'utilisation d'une nouvelle bibliothèque (dlib) pour la détection du visage et l'extraction des points de saillances. Ensuite, l'application du classifieur bayésien pour détecter trois émotions (joie, colère et surprise). Le model a été validé par la validation croisée a k-plis avec une précision de 82%, 97% et 31% pour la surprise, la joie et la colère respectivement.

Le projet le plus récent est celui de J. NOWAK et S. RIMLINGER [6]. Il traite la détection du visage et l'extraction des points de saillances dont une grande partie était consacrée à l'amélioration des performances et l'évaluation de la détection, de la bibliothèque dlib, sur les différentes contraintes (l'intensité de lumière, l'angle de rotation...). Ils ont clôturé leur réalisation par la mise en place d'une méthode de détection d'ouverture de bouche.

1.3 Problématique

Plusieurs sujets ont été abordé dans les travaux précédents du projet Psyphine. Cependant, la diversité et la richesse des disciplines du projet Psyphine ont rendu difficile de travailler sur tous les axes du projet. Par conséquence, et puisque le but du projet est d'étudier les réactions et le comportement des gens à travers des expériences d'une durée de 10 minutes dans laquelle Ils seront en interaction non verbale avec la lampe rebotisé.

Notre contribution est donc, d'affecter à la lampe la capacité d'interagir (de se comporter) conformément au changement d'expressions du visage de la personne en face. En effet, problématique principale est comment concevons-nous un système qui permet à la lampe d'exécuter des comportements suivant les différentes réactions des gens perçues par le biais de sa webcam.

Nous pouvons diviser cette problématique en deux sous problématiques dont la première sera le choix de la méthode d'extraction et la représentation des comportements des gens et la deuxième sera le regroupement (clustering) de ces comportements afin de décider le comportement approprié.

Conclusion

Ce chapitre à servie comme introduction au projet et à notre problématique sur laquelle notre contribution sera basée. Dans le chapitre suivant nous plongeons dans les notions techniques des méthodes existantes dans l'état de l'art.

Chapitre 2

État de l'art

« ...ce que nous voulons, c'est une machine qui peut apprendre de l'expérience ». A. Turing.

Sommaire

Introduction **9**
2.1 Méthodes de détection de visage **9**
2.1.1 Détection avec la méthode Haar cascade 9
2.1.2 Détection avec les réseaux de neurones profonds (DNN) 10
2.1.3 Détection avec la l'Histogrammes de gradients orientées (HOG) 10
2.1.4 Détection avec les réseaux de neurones convolutionnel (CNN) . . 12
2.2 Methode d'extraction des points de saillances **13**
2.3 Clustering **15**
2.3.1 K-means 15
2.3.2 Carte auto-organisatrice (SOM) 16
Conclusion **18**

Introduction

Dans le but de faciliter la compréhension de notre solution dans le chapitre prochain, nous introduisons dans ce chapitre les differentes notions de base employé, ainsi, nous présentons les méthodes existantes pour les differents phases, de la détection du visage jusqu'à la classification non supervisé.

2.1 Méthodes de détection de visage

La détection de visage est un type d'application classé dans la technologie de vision par ordinateur. C'est le processus aux cours duquel des algorithmes sont développés et formés pour localiser correctement les visages ou les objets. Celles-ci peuvent être en tems réel à partir d'une caméra vidéo. Afin de détecter la position des visages dans les images de sorte à obtenir une région d'intérêt sur laquelle l'extraction des vecteurs de caractéristiques pourra être accomplie il y a bien de nombreuses méthodes, dans ce qui suit nous allons présenter les méthodes les plus utilisées dans la détection.

2.1.1 Détection avec la méthode Haar cascade

Les cascades de Haar est une méthode utilisée pour la détection d'objet de classificateurs en cascade basés sur les fonctionnalités Haar, c'est une méthode efficace de détection d'objets proposée par Paul Viola et Michel Jones en 2001. Il s'agit d'une approche basée sur l'apprentissage automatique.

La fonction cascade est formée à partir de nombreuses images positives et négatives. Il est par la suite utilisé pour détecter des objets dans d'autres images. Cette méthode travaille avec la détection de visage. Initialement, l'algorithme nécessite beaucoup d'images positives (image de visage) et d'images négatives (images sans visages) pour former le classifieur. Ensuite, extraire les caractéristiques, et pour cela, les fonctionnalités de Haar présentées dans la figure ci-dessous sont utilisées. Chaque caractéristique est une valeur unique obtenue par la soustraction de la somme des pixels sous le rectangle blanc de la somme des pixels sous le rectangle noir.

(a) Caractéristiques du bord (b) Caractéristiques de ligne (c) Caractéristiques quatre rectangles

FIGURE 2.1 – Caractéristiques de Haar

Cette méthode consiste à parcourir l'intégralité de l'image pour extraire un certain nombre de caractéristiques comme ça permet de réduire le nombre de calculs relatif à un pixel donné à une opération ne nécessitant que quatre pixels. Parmi les caractéristiques

calculées, il y'en a celles qui sont sans importance, pour sélectionner que le nombre de caractéristiques importantes Haar Cascade utilise l'algorithme d'apprentissage Adaboost pour à la fin obtenir un résultat efficace des classifieurs.

Une fois le visage détecté, il faut en extraire les saillances. L'utilisation des caractéristiques rectangulaires de cette méthode permet de détecter les yeux ou la bouche sur un visage, et cette fonction est disponible dans OpenCV. Cette méthode a résolue beaucoup de problèmes rencontrés auparavant dans la détection de visage tel que, cette méthode fonctionne presque en temps réel sur le processeur, elle en dispose d'une architecture simple et elle détecte les visages à différentes échelles. Cependant cette méthode reste imprécise, elle donne beaucoup de fausses prédictions, elle ne fonctionne pas sur les images non frontales et elle ne fonctionne pas sous occlusion.

2.1.2 Détection avec les réseaux de neurones profonds (DNN)

La méthode DNN ou Deep Neural Network (réseau de neurones profond) est une méthode de détection de visage plus performante basée sur l'apprentissage profond aussi appelé deep-learning [3]. Elle repose sur l'apprentissage, par un ordinateur d'un modèle de données servant à remplir une tâche. Pour ce faire, un réseau de neurones est modélisé dans l'ordinateur. Celui-ci reçoit des images les traite en vue de leur classification par le biais de modèle de l'apprentissage profond formé. Cette méthode donne en sortie une matrice à quatre dimensions telle que, la troisième dimension itère sur les visages détectés tandis que la quatrième dimension contient des informations sur le cadre de la sélection et le score de chaque visage. Les coordonnées de la sortie du cadre de sélection sont normalisées entre [0,1]. Ainsi, les coordonnées doivent être multipliées par la hauteur et la largeur de l'image d'origine pour obtenir le cadre de sélection correct sur l'image. La méthode DNN surmonte tous les défauts de la détection avec la méthode à cascade de Haar, sans compromettre aucun avantage fourni par Haar. En plus de ça, c'est la méthode la plus précise dans la détection, elle fonctionne en temps réel sur le processeur ainsi elle détecte les visages à différentes échelles (visage grand et petit) et elle fonctionne pour les différentes orientations du visage (haut, bas, gauche, droite, côté, etc) voire même sous occlusion importante. Actuellement le méthode DNN ne présente aucun inconvénient majeur d'après les dernières recherches faites en fin 2018 par rapport à toutes les méthodes de détection de visage dans OpenCV sauf qu'elle est plus au moins lente que celle basée sur Dlib HOG.

2.1.3 Détection avec la l'Histogrammes de gradients orientées (HOG)

Il s'agit d'un modèle de détection de visage largement utilisé, inventé en 2005, basé sur les fonctionnalités HOG (Histogram of Oriented Gradients) et SVM (Support Vector Machine). Dans le descripteur de caractéristiques HOG, la distribution (histogrammes) des directions de gradients (gradients orientés) est utilisé comme caractéristique. Le calcul de l'histogramme des gradients se déroule en différentes étapes.

Tout abord, nous allons mettre l'image en noir et blanc, nous enlevons toutes les autres couleurs car nous n'avons pas besoins de couleurs pour trouver un visage. Ensuite, sélectionner un patch d'image de l'image initiale et redimensionner ce patch de l'image à une

taille plus petite dans le but de regarder chaque pixel de notre image. Pour chaque pixel, nous voulons regarder les pixels qui l'entourent directement. Le but est de déterminer l'obscurité du pixel courant par rapport aux pixels qui l'entourent. Ensuite dessiner une flèche pour montrer dans quelle direction l'image devient plus sombre comme le montre la figure ci-dessous :

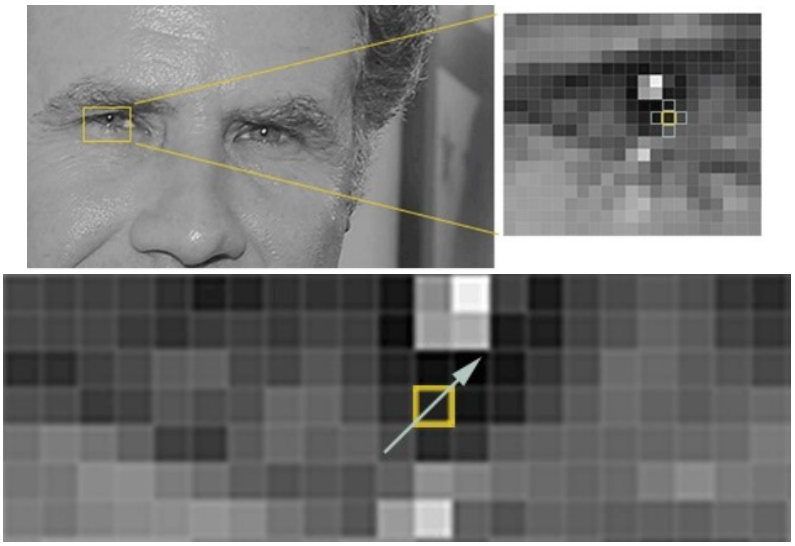


FIGURE 2.2 – construction du gradient orienté

En répétant ce processus, chaque pixel sera remplacé par une flèche. Ces flèches sont appelées gradients et elles montrent l'écoulement de la lumière à l'obscurité sur toute l'image.

La raison majeure pour laquelle cette méthode remplace les pixels par des flèches est pour avoir une représentation exacte des images. En analysant directement les pixels, pour la même personne, les images vraiment sombres et les images vraiment claires auront des valeurs de pixels totalement différentes. En revanche, en considérant seulement la direction dans laquelle la direction change, nous arrivons à une représentation exacte.

Cependant, sauvegarder le gradient pour chaque pixel nous donne beaucoup de détails et cela fait beaucoup de données, pour cela, nous allons découper l'image en petits carrés de 16x16 pixels chacun. Et pout chaque case, on comptera combien de pentes dans chaque direction principale (vers le haut, haut droit, haut gauche, etc.. . .). Ensuite, nous remplaçons ce carré dans l'image par la direction des flèches. A la fin, on arrive à transformer l'image originale en une représentation très simple qui capture la structure de base d'un visage de manière très simple.

A la fin, on calcule le vecteur de caractéristiques HOG, ce dernier est un géant vecteur qui concatène plusieurs vecteurs de caractéristique de l'ensemble de l'image pour chaque carré de cette image comme il est illustré dans la figure 2.3 ci-aprés.

La dernière étape consiste à calculer le vecteur de caractéristique HOG, pour calculer le dernier vecteur de caractéristique pour l'ensemble du bloc d'image, les vecteurs de 36 x 1 sont concaténés en un seul vecteur géant.

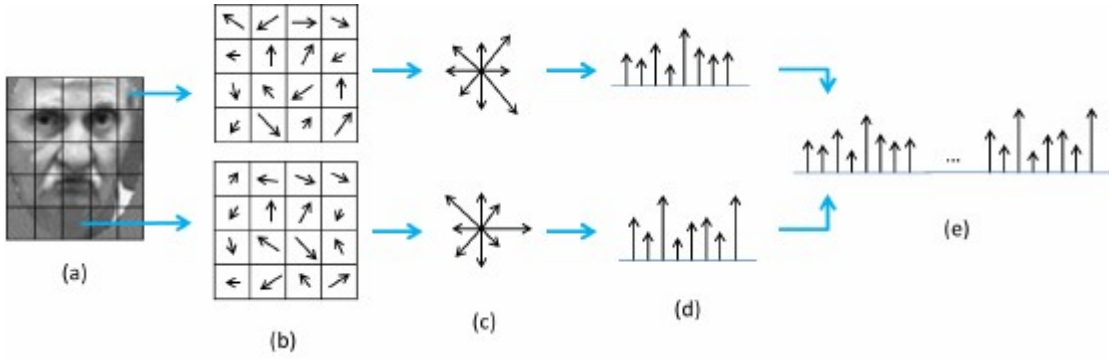


FIGURE 2.3 – Processus de la construction de l'histogramme des gradients

La méthode HOG est la méthode la plus rapide sur le processeur, elle fonctionne très bien pour mes faces frontales et légèrement non frontales, son modèle est léger par rapport aux méthodes précédentes et elle fonctionne sous petit occlusion. Cependant, l'inconvénient majeur de cette méthode est qu'elle ne détecte pas les petits visages, car elle est conçue pour une taille minimale de 80 x 80, la boite englobante exclut souvent une partie du front et même une partie menton parfois, elle ne fonctionne pas très bien sous occlusion importante et elle ne fonctionne pas pour les faces latérales et les faces extrêmes non frontales, comme par exemple regarder vers le bas ou vers le haut.

2.1.4 Détection avec les réseaux de neurones convolutionnel (CNN)

C'est une méthode qui est aussi très utilisée dans cette dernière décennie, à cause de sa capacité et la perfection dans l'extraction qui a permet aux utilisateurs de se libérer des tâches fastidieuses telles que l'extraction minière négative. Cette méthode repose sur un algorithme de détection d'objets à marge maximale MMOD (Max Margin Object Detection en anglais) Pour Dlib. Ce modèle produit des détecteurs de haute qualité à partir de quantités relativement faibles de donnée d'entrainement (à partir de 4 images seulement) en utilisant un ensemble de données étiqueté manuellement par son auteur. Cependant, l'implémentation MMOD utilise l'extraction de caractéristiques HOG suivie d'un filtre linéaire unique. Cela signifie qu'il capable d'apprendre à détecter des objets présentant une variation complexe de pose ou une grande variété d'apparences, ce qui a conduit à utiliser au cours des dernières années, les réseaux de neurones convolutifs CNN [3] capable de traiter tous ces problèmes dans un seul modèle. L'idée est donc d'ajouter une implémentation de MMOD avec l'extraction de la fonctionnalité HOG remplacée par un réseau de neurones convolutifs. Le résultat obtenu avec la version CNN de MMOD était inattendu en travaillant seulement avec 4 images étant donné que d'autres méthodes de Deep-learning nécessitent généralement plusieurs milliers d'images. Les images suivantes montrent la différence de capacité dans la détection entre le nouveau détecteur CNN et le détecteur de Dlib par défaut HOG telles que, les cases rouges correspondent aux détections CNN et les cases bleues aux détections HOG.

Une différence notable entre les deux modèles de détection, HOG fait un excellent travail sur les visages faciles en regardant la caméra, mais seulement en étant directement

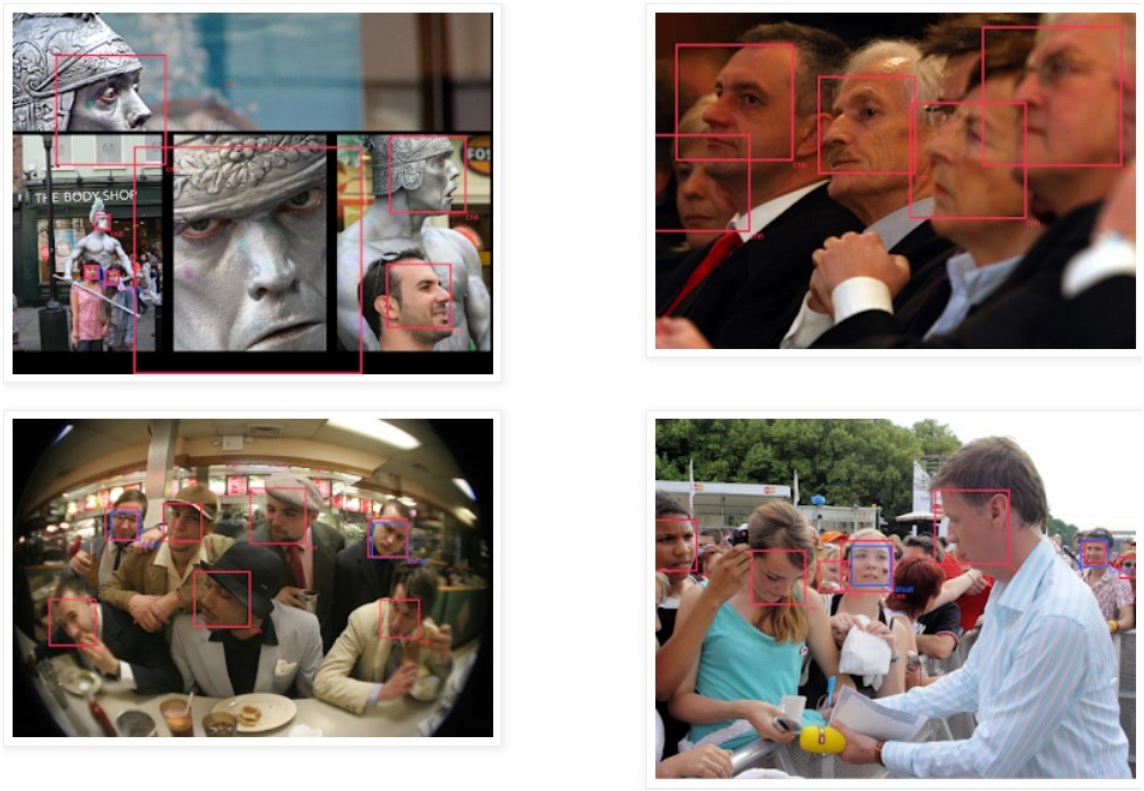


FIGURE 2.4 – visages détectés par le détecteur HOG et CNN

face à la caméra, cependant e détecteur CNN est bien meilleur non seulement pour traiter les cas facile mais tous les visages en général. Il est aussi robuste à l'occlusion, rapide sur les GPUs (45 ms par images) et son processus de formation est très facile. En revanche, ce modèle reste lent sur mes processus (370 ms pour traiter une seule image), il entrainé pour une taille minimale de 80 x 80, ce qui fait qu'il ne détecte pas les petits visages et sa boite englobante est encire plus petite que le détecteur HOG.

2.2 Methode d'extraction des points de saillances

Le processus qui est capable d'explorer un ensemble de points clés à partir d'une image de visage donnée, est appelée Localisation du repère du visage ou Face Landmark Localization en anglais (alignement du visage).

Les repères (points clés) qui nous intéressent sont ceux qui décrivent la forme des attributs du visage comme : les yeux, les sourcils, le nez, la bouche, et le menton. Ces points ont donné un excellent aperçu de la structure faciale analysée, qui peut être très utile pour un large éventail d'applications.

De nombreuses méthodes permettent de détecter ces points : certaines d'entres elles atteignent une précision et une robuste supérieures en analysant un modèle de visage 3D extrait d'une image 2D, d'autres s'appuient sur la puissance des CNN et d'autres utilisent

des fonctions simples (mais rapides) pour estimer l'emplacement des points.

L'Algorithme de détection des repères de visage reposé par Dlib est l'implémentation de l'Ensemble of Regression Trees (ERT) présenté en 2014 par Kazemi et Sullivan [7]. Cette technique utilise une fonction simple et rapide (différence d'intensité des pixels) pour estimer directement la position des points de repère. Ces positions estimées sont ensuite affinées au moyen d'un processus itératif effectué par une cascade de variables explicatives. Les régresseurs produisent une nouvelle estimation à partir de la précédente, en essayant de réduire l'erreur d'alignement des points estimés à chaque itération. L'algorithme est très rapide, en fait, il faut environ 1 à 3 ms pour détecter un ensemble de 68 points de repère sur un visage donné. La figure suivante présente l'ensemble des points de visage extrait par l'algorithme :

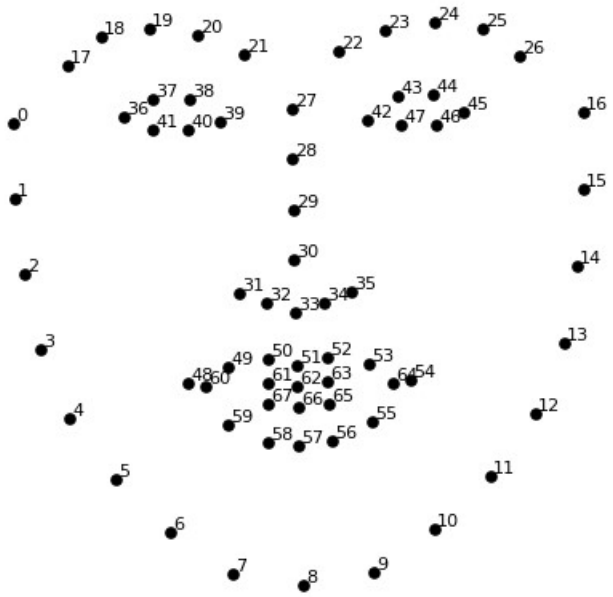


FIGURE 2.5 – L'ensemble des 68 points détectés par Dlib pré-entraîné

Fondamentalement, un prédicteur de forme peut être généré à partir d'un ensemble d'images, d'annotations et d'options de formation. Une seule annotation se compose de la région du visage et des points marqués que nous voulons localiser. La région du visage peut être facilement obtenue par n'importe quel algorithme de détection de visage (OpenCV Haar Cascade, Dlib HOG Detector, CNN, ...), au lieu de cela les points doivent être marqués manuellement ou détectés par des détecteurs et modèles déjà disponible. Enfin, les options de formation sont un ensemble de paramètres qui définissent les caractéristiques du modèle formé. Ces paramètres peuvent être correctement ajustés afin d'obtenir le comportement souhaité du modèle généré.

2.3 Clustering

Le clustering ou la classification non supervisée en français, est une technique de classification très parlante en apprentissage automatique, en analyse et en fouille de donnée ainsi qu'en reconnaissance de formes. Il fait une partie intégrante de tout un processus d'analyse exploratoire de données permettant de produire ses outils de synthétisation, de prédiction, de visualisation et d'interprétation d'un ensemble d'individus (personnes, objets, processus, etc.). L'objectif est, à partir de données constituées d'un ensemble d'individus ou d'objets et d'une relation de proximité entre ceux-ci, de construire des groupes d'individus homogènes dans les sens où :

— Deux individus proches doivent appartenir à un même ensemble (groupe).
— Deux individus éloignés doivent appartenir à des groupes différents.

Afin de définir l'homogénéité d'un groupe d'observation, il est nécessaire de mesurer une ressemblance entre deux observations. D'où la notion de similarité et de dissimilarité [9].

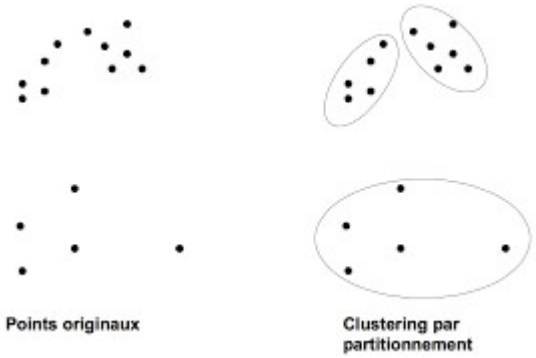


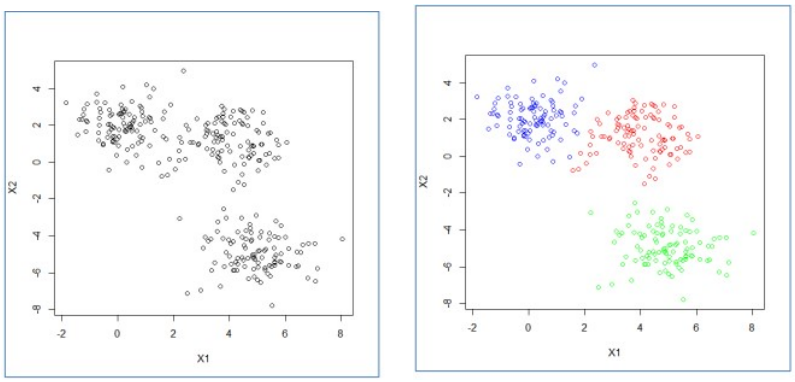
FIGURE 2.6 – Clustering

Ici, les points originaux ce sont les données constituées par exemple un ensemble d'individus, le clustering les partitionne dans des groupes où chaque groupe représente un ensemble d'individus qui partagent une même caractéristique. Le clustering comprend plusieurs algorithmes de classifications pour classifier chaque point de données dans un groupe spécifique tels que : k-Means le plus algorithme connu dans la classification non supervisée, Mean-Shift un algorithme basé sur une fenêtre glissante qui tente de trouver des zones senses de points de données, Density-Based Spatial Clustering of Applications with Noise (DBSCAN), un algorithme basé sue la densité similaire au décalage moyen, Expectation-Maximization (EM) à l'aide de Modèles de Mélange Gaussiens (GMM) et Self Organisation Map (SOM). Dans ce qui suit nous allons nous intéressés aux deux algorithmes suivent k-Means et l'algorithme Safe Organizen Map (SOM).

2.3.1 K-means

K-Means est probablement l'algorithme de classification le plus connu. C'est un algorithme facile à comprendre et à implémenter.

La figure ci-dessus montre un exemple de classification par l'algorithme k-Means, tels que les points noirs à droites sont les points de données de départs et à la fin on affecte



(a) avant le clustering (b) aprés le clustering

FIGURE 2.7 – K-means clustering

chaque point dans son groupe, ici nous avons trois groupes (groupes des points bleus, verts et rouges).

Pour commencer, nous avons d'abord sélectionnés un certain nombre de classes (groupes) à utiliser en initialisant aléatoirement leurs points centraux respectifs. Dans notre exemple nous avons trois classes. Les points centraux sont des vecteurs de la même longueur que chaque vecteur de points de données.

Chaque point de données (points noirs) est classé en calculant la distance entre ce point et chaque centre de groupe, puis en classant le point dans le groupe dont le centre est le plus proche. Sur la base de ces points classés, nous recalculons le centre du groupe en prenant la moyenne de tous les vecteurs du groupe.

Nous répétons cette étape pour un nombre défini d'itérations ou jusqu'à ce que les centres de groupe ne changent plus beaucoup d'une itération à une autre. Comme il est possible de choisir d'initialiser plusieurs fois le centre de groupe de manière aléatoire, puis de sélectionner le cycle qui donne les meilleurs résultats.

K-Means a l'avantage d'être assez rapide, car nous ne faisons que calculer les distances entre les points et les centres de groupe. Très peu de calculs, sa complexité linéaire est O (n).

D'autres parts, K-Means présente quelques inconvénients. Tout d'abord, nous devons sélectionner le nombre de groupes (classes). Ce n'est pas toujours évident et idéalement avec un algorithme de classification, nous aimerions qu'il soit mieux compris pas ceux-ci, car il s'agit là d'obtenir un aperçu des données. K-Means commence également par un choix aléatoire de centre de grappes et peut donc donner différents résultats de frappes dur différentes exécutions de l'algorithme. Ainsi, les résultats peuvent ne pas être reproductible et manquer de cohérences.

2.3.2 Carte auto-organisatrice (SOM)

Self Organizen Map ou la carte auto organisatrice (SOM), est un type de réseau de neurones artificiels (RNA) formé à l'aide d'un apprentissage non supervisé afin de pro-

duire une représentation discrète, généralement bidimensionnelle, de l'espace d'entrée des échantillons d'apprentissage. La carte est donc une méthode pour faire la réduction de dimensionnalité. Les carte auto organisatrices différent des autres réseaux de neurones artificiels par le fait qu'elles appliquent l'apprentissage compétitif par opposition à l'apprentissage avec correction d'erreur et qu'elles utilisent une fonction de voisinage pour préserver les propriétés topologiques de l'espace d'entrée.

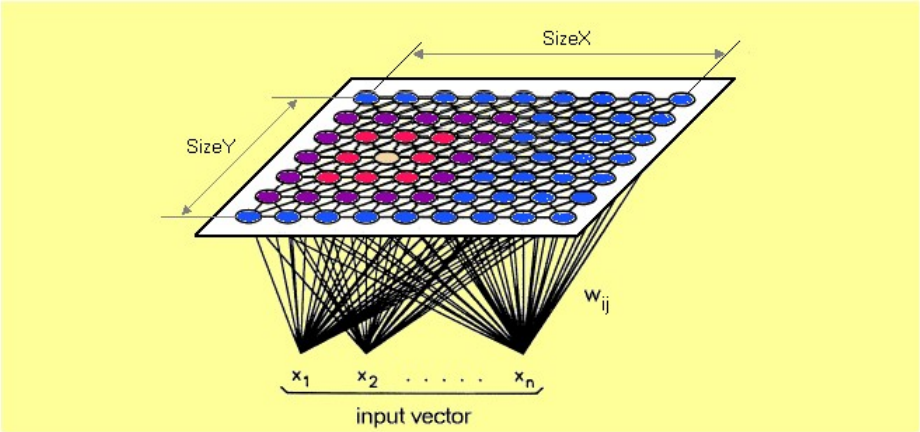


FIGURE 2.8 – Réduction de la dimensionnalité dans la SOM

SOM a été introduite par le professeur finlandais Teuvo Kohonen dans les années 1980, elle est aussi appelée carte de Kohonen [8].

Chaque point de données dans l'ensemble des points de données se reconnait en compétition pour la représentation. Les étapes de mappage SOM commencent par l'initialisation des vecteurs de pondération. A partir de là, un vecteur d'échantillon est sélectionné de manière aléatoire et la carte des vecteurs de poids est explorée pour trouver quel poids représente le mieux cet échantillon. Chaque vecteur de poids a des poids voisins qui lui sont proches. Le poids choisi est récompensé par sa capacité de ressembler davantage à cet échantillon de vecteur sélectionné au hasard. Les voisins de ce poids sont également récompensés par leur capacité à ressembler davantage au vecteur échantillon choisi. Cela permet à la carte de s'agrandir et de former différentes formes. Plus généralement, ils forment des formes carrées, rectangulaires, hexagonales et L dans un espace de fonction 2D.

L'algorithme SOM :

1. Initialiser chaque poids de chaque nœud.
2. Choisir un vecteur au hasard dans l'ensemble des données d'apprentissages.
3. Chaque nœud est examiné pour calculer les poids qui ressemblent le plus au vecteur d'entrée. Le nœud gagnant est généralement appelé unité de meilleure correspondance ou Best Matching Unit (BMU)
4. Le poids gagnant est récompensé par le fait de ressembler davantage au vecteur échantillon. Les voisins deviennent également davantage comme le vecteur échan-

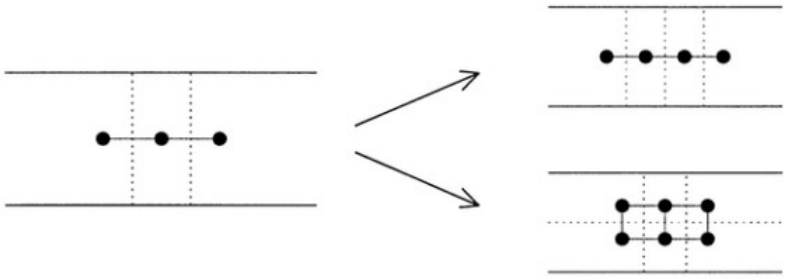


FIGURE 2.9 – Illustration de la décision de base à prendre lors de la croissance

tillon. Plus le nœud est proche du BMU, plus des poids sont modifiés et plus le voisin est éloigné du BMU, moins il en apprend.

5. Répéter l'étape 2 pour N itération.

BMU est technique qui calcul la distance entre chaque poids et le vecteur échantillon en parcourant tous les vecteurs de poids. Le poids avec la distance la plus courte est le gagnant. Il existe nombreuse façon de déterminer la distance, cependant la méthode la plus couramment utilisée est la distance euclidienne.

Conclusion

Nous avons exploré, dans ce chapitre, les différents notions, existantes, liées à notre solution. Cela nous permet d'entamer les détails de notre conception dans le chapitre suivant.

Chapitre 3

Analyse et réalisation

« ...si vous ne pouvez pas l'expliquer simplement, vous ne le comprenez pas assez bien. ». A. Einstein.

Sommaire

Introduction 20
3.1 Appoche globale 20
3.2 Phase de préparation 21
3.2.1 Prétraitement 21
3.2.2 Détection du visage 21
3.2.3 Éxtraction des points de saillances 22
3.3 Phase d'extraction des VCC 22
3.3.1 Éxtraction et encodagedes caractéristique 22
3.3.2 Vecteurs caractéristiques des changements d'expressions 23
3.4 Phase de décision 24
3.4.1 K-Means 24
3.4.2 Carte auto-organisatrice dynamique 25
Conclusion 26

Introduction

Les deux premiers chapitres étaient une introduction au contexte du projet ainsi q'une présentation de toutes les notions importantes pour la compréhension de l'approche.

Dans ce chapitre, nous présentrons les différentes étapes de l'approche suivie en expliquant ce que nous avons utilisé dans chacune de ces étapes.

3.1 Appoche globale

Notre objectif, comme mentionné précédemment, est de concevoir un système qui permet à la lampe de réagir en fonction des changements des expressions du sujet qui se trouve en face d'elle.

La réalisation de ce système nécessite de passer par plusieurs étapes indispensables en commençant par la détection de visage et l'extraction des points de saillances. Ensuite nous allons passer à l'extraction des caractéristiques et l'encodage des solutions. Enfin, nous arrivons au regroupement (clustering) des différents changements et à la définition du comportement.

Pour simplifier la compréhension du fonctionnement de l'approche suivie, nous avons jugé utile de partitionner le travail en deux phases (figure 3.1) :

— La phase de préparation qui prend en entrée un flux d'image sous forme d'une vidéo (séquences d'images) pour produire en sortie un vecteur caractérisant les changements sur les différentes séquences.

— La deuxième phase, dite d'extraction des vecteurs caracteristiques des changements VCC, comporte deux étapes : l'extraction des caracterisitques, puis, le calcule des vecteurs caracteristiques des changements d'expressions.

— La derniere phase est la phase de décision qui consiste à regrouper les vecteurs caractéristiques en entrée afin de décider le comportement à exécuter (les commandes à envoyer à la lampe). Nous verrons ses phases en détail dans les sections qui suit.

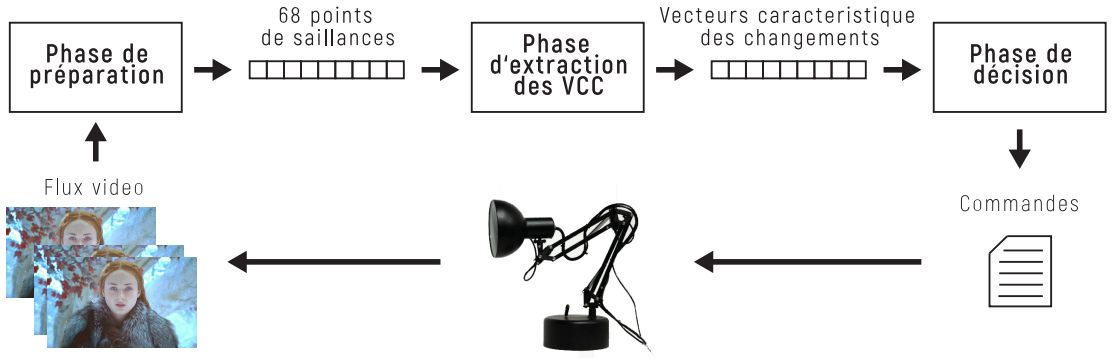


FIGURE 3.1 – Structure globale du systèùme

3.2 Phase de préparation

Dans cette section nous entamons la phase de préparation, cette phase a pour but d'extraire un vecteur de points de saillances pour les faire passer à la phase suivante.

3.2.1 Prétraitement

Le prétraitement est l'étape avant-première (ou l'étape zéro), dans laquelle nous récupérons les images RGB de flux vidéo. Ensuite, nous les convertirons en niveaux de gris (grayscale) car nous n'avons pas besoin des couleurs dans nos prochains traitements (détection du visage, extraction des points de saillances...). Ces images seront ainsi redimensionnées pour diminuer le temps des traitements.

Afin d'obtenir de bons résultats dans les prochaines étapes, nous avons jugé utile d'appliquer une égalisation d'histogramme en niveaux de gris. Cette dernière permet de mieux répartir les intensités des images à faible contraste pour obtenir des images de meilleure qualité. Un résumé de l'étape de prétraitement est illustré par la figure suivante.

FIGURE 3.2 – Prétraitement

3.2.2 Détection du visage

Comme illustré par la figure 3.3 ci-dessous, l'étape de détection du visage s'agit de retrouver la zone du visage (la région d'interet ROI) dans une image pour qu'on puisse y effectuer des traitements par la suite.

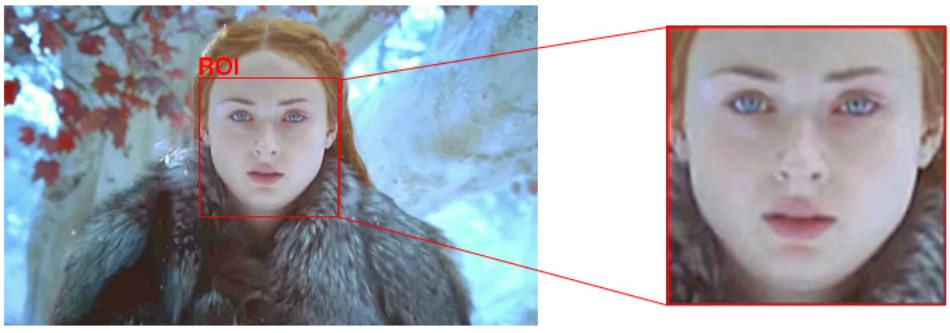


FIGURE 3.3 – Détection de visage

La méthode utilisée pour cette étape est l'histogramme de gradient orienté (HOG) décrit dans (section 2.1.3). Nous avons utiliser son implémentation dans la bibliothèque dlib pour extraire la région d'interet des images de flux video.

3.2.3 Éxtraction des points de saillances

Une fois l'étape de détection de visage est terminée, l'étape d'extraction des points de saillances prend place. Cette étape utilise la région (ROI) extraite de l'étape précédente pour déterminer 68 points de saillances. Nous allons utiliser un algorithme dit « algorithme d'estimations des points de saillances » qui est basé sur une approche inventée par Vahid Kazemi et Josephine Sullivan en 2014 [7] basé sur des arbres de regressions (un type d'arbre de décision).

FIGURE 3.4 – Éxtraction des points de saillances

L'idée de base est de définir 68 points spécifiques (appelés repères) qui existent sur chaque visage - le haut du menton, le bord extérieur de chaque œil, le bord intérieur de chaque sourcil, etc. Ensuite, utiliser un algorithme d'apprentissage automatique pour avoir un model capable de trouver ces 68 points sur n'importe quel visage.

3.3 Phase d'extraction des VCC

Cette phase permet en premier, de définir des caracteristique d'expressions ,choisis explicitement, à partir des vecteurs de points de saillances. Ensuite, utiliser ces caractéristiques pour calculer les vecteurs caractéristiques des changements d'expressions qui seront utilisé pour le clustering dans la phase qui suit.

3.3.1 Éxtraction et encodagedes caractéristique

Effectuer un apprentissage sur tous les 68 points extraits est, d'une part, trés couteux en terme du temps et d'espace ainsi, il peuvent générer de bruit qui affecte la qualité des résultat obtenus. Par conséquent, nous allons extraire, à partir de ces points, des caracteristiques qui nous permettent de destinguer les differentes expressions du visage. Nous avons consideré six caracteristiques (à savoir : distance des sourcils, ouverture des

yeux, position du visage, ouverture de la bouche et la rotation du visage), ils sont illustrés sur la figure ci-dessous.

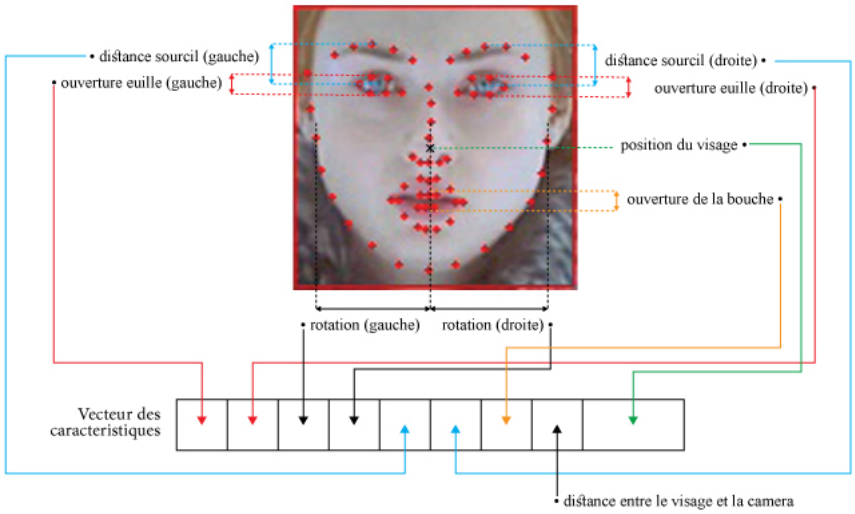


FIGURE 3.5 – Éxtraction des caractéristiques

Dans notre experience, la lampe rebotisé est mis devant differents personnes voire les mêmes perssonnes peuvent se retrouver à des distances differents de la lampe. Cela engendre une diversité de valeurs des caracteristiques définis ci-dessus pour chaque personne et aussi pour la meme personne avec des differents distances. De ce fait, il est nécéssaire de normaliser ce vecteur des caracteristique pour que la diversité des personnes (ou leurs distances de la camera) n'affecte pas les résultat retournés. Nous avons donc normaliser notre vecteur caracteristique, en prenant le rapports des caracteristiques sur des distance fixe dans le visage, par exemple le rapport entre la largeur et la longueur des yeux represente le proportion d'ouverture des yeux. En conséquence, la valeur de chacune des caracteristique est compris entre 0 et 1.

3.3.2 Vecteurs caractéristiques des changements d'expressions

Jusqu'à la, nous avons pu extraire les points des saillances et en définir des caracteristique avec. Donc on peut passer ces vecteurs pour la prochaine phase pour le clustering des expressions. Cependant, nous nous intéressons du *chagement des expressions* et non plus des expressions uniques. Pour cela, il faut traiter plusieurs images (séquence d'images) pour qu'on puisse calculer le vecteur caractérisant les changements d'expressions sur cette séquence.

Une approche simple pour le faire, est de prendre les images du flux video deux à deux. Puis, on calcule le changement entre ces deux images. Soientt deux images i et j, extraites de flux video dans deux instant diffèrents tel que $temps(i) < temps(j)$. D'abord, les étapes de détéction du visage, d'éxtraction des points de saillances et d'extraction des caracteristiques sont effectués sur ces deux images pour avoir les vecteurs caracteristiques

Vi (respectivement Vj) des images i (respectivement j). Ensuite, nous calculons le *vecteur caracteristique des changments* en appliquant une simple soustraction $Vj - Vi$ sauf pour la position, nous calculons la distance euclidienne. Nous obtenons finalement un vecteur caracterisant les chagments entre entre l'image i et l'image j.

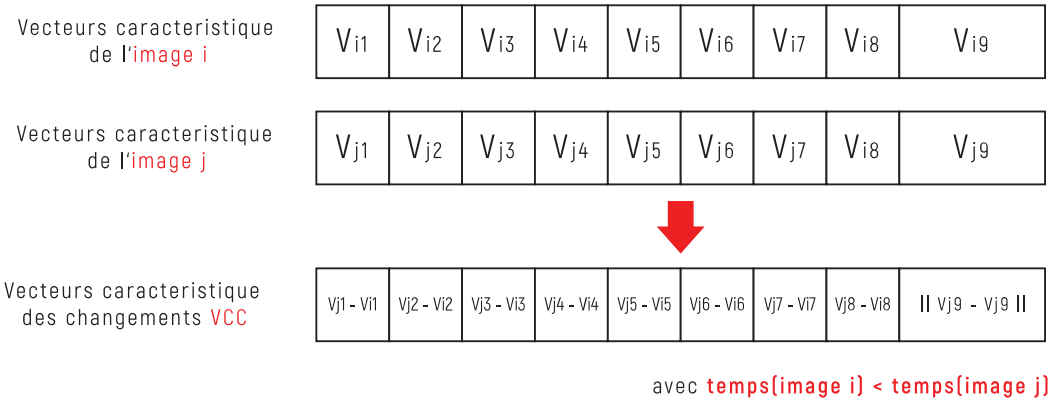


FIGURE 3.6 – Calcul des vecteurs caractèristiques des changements

3.4 Phase de décision

La dernière phase de l'approche, est celle de décision, dans laquelle nous prenons les vecteurs caractéristiques issues de la phase antérieure, et on y applique des algorithmes d'apprentissage pour avoir en sortie le comportement approprié.

Nous pouvons formuler cela sous forme d'un problème d'apprentissage non supervisé discret où les des données sont les vecteurs caractéristiques des changements d'expresions et les clusters, ou encore classes ou groupes, représente les comportements. En effet, nous devons entrainer un model f (une fonction de X dans Y) qui permet de retourner pour chaque entrée $x \in X$ le comportement approprié $y \in Y$:

$$f : X \to Y$$

Avec X ensemble des vecteurs caractéristiques des changements et Y l'ensemble des comportements.

De plus, l'algorithme doit être capable d'apprendre en temps réel en raffinant le model à chaque nouvelle donnée d'entrée. Ce qui rend la tache plus complexe. Pour ce faire, nous avons décider de mettre en place deux modéles, une variante du k-moyennes (k-means) et une variante de la carte auto-organisatrice.

3.4.1 K-Means

L'algorithme de base de k-means (expliqué dans le chapitre précedent section 2.3.1) prend en entrée une base de vecteurs d'entrée et le nombre clusters qu'ils représente dans notre cas d'étude les comportements souhaités. Cependant, notre application est en temps réel, c.-à-d. nous aurons une nouvelle entrée chaque intervalle du temps, tout au long du

flux vidéo. De plus, le model doit être capable de s'adapter avec les différentes entrées à n'importe quel instant. L'utilisation de l'algorithme de base pour l'application n'est pas le bon choix, car il nécessite un espace mémoire important due à la l'extension de la base en temps réel, et les temps d'exécution augmente en fonction de la taille de la base ce qui résulte un temps de réponse très long.

Pour pallier ce problème, nous avons pensé en premier à une approche qui consiste à construire une base, chaque une période du temps, et la passer à l'algorithme. Cette approche permet d'économiser l'espace mémoire en conservant qu'une petite base chaque période. Néanmoins, on aura un nouveau modèle pour chaque base diffèrent totalement au modèle qui le précède car les bases d'entrées sont différentes.

Les inconvénients de l'approche précédente nous ont amené à penser à une deuxième approche plus performante. Cette dernière, et comme illustré par la figure 3.7, commence par la collection d'une base initial de taille n prédéfinie, ensuite, passer cette base pour l'algorithme k-moyennes qui permet de produire un model initial. Par la suite, nous gardons en mémoire que les poids des prototypes des classes (les points centres représentant les clusters), puis, nous raffinons ce modèle pour chaque vecteur d'entrée.

FIGURE 3.7 – Variante de l'algorithme k-moyennes

3.4.2 Carte auto-organisatrice dynamique

Nous avons vue en chapitre deux, la carte organisatrice de Kohonen et son fonctionnent. Pour notre problème, nous utilisons une des variantes des cartes auto-organisatrices bi-dimensionnelle dite la carte auto-organisatrice dynamique (en anglais Dynamic Self Organizing Map - DSOM) proposé par Nicolas P. Rougier et Yann Boniface dans [12].

Au contraire de l'algorithme original de la carte organisatrice qui est dépendant du temps (taux d'apprentissage et voisinage), celui de DSOM est invariant dans le temps. Cela permet un apprentissage en ligne et continu sur les distributions de données statiques et dynamiques. De plus, la densité obtenue par cette variante n'est pas directement proportionnelle à la densité de la distribution [12].

La figure ci-dessus montrent la structure de la carte auto-organisatrice choisie pour concrétiser ce problème nous pouvons observer que les entrées et représentent nos vecteurs caractéristiques des changements et la sortie est une matrice de neurones dans chacun représente un comportement bien défini. Le neurone est représenté par un vecteur des poids de taille des entrées. Ces poids seront modifiés par apprentissage tout au long d'exécution.

Au départ, nous initialisons le poids des neurones aléatoirement. Ensuite pour chaque vecteur caractéristique des changements en entrée, nous récupérons d'abord les comportements appropriés à ce dernier en prenant le neurone qui a la plus petite distance (le meilleur match BMU). Puis, nous utilisons ce vecteur d'entrée et le BMU pour ajuster les poids des neurones. Cela correspond à l'entraînement de la DSOM. En effet, les poids de BMU et son voisinage, determiné par la fonction gaussienne, seront modifiés par la formule suivante [12] :

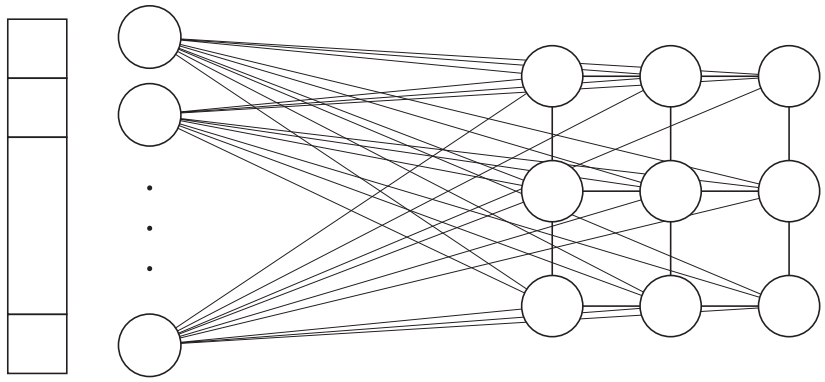


FIGURE 3.8 – Structure de la DSOM utilisée

$$\Delta w_i = \varepsilon \parallel v - w_i \parallel h_\eta(i, s, v)(v - w_i)$$

Avec :
- Δw_i : le nouveau poids et w_i l'ancien poids
- v : le vecteur d'entrée
- ε : est la constante du taux d'apprentissage (learning rate)
- $h_\eta(i, s, v)$: la fonction du voisinage où
  - i l'indice du neurone
  - s représente le meilleure correspondant (BMU)
  - η l'élasticité (un seuil pour considéré qu'un neurone et suffisament proche pour representer les données et donc les poids ne seront pas modifiés)

Dans le cas où le neurone est suffisamment proche de la nouvelle entrée (distance $<$ seuil), nous ne modifierons pas les poids car ce neurone est considéré comme représentant optimale du groupe dont cette donnée appartient. Nous répétons ces étapes tous le temps de l'exécution.

Conclusion

Nous avons présenté, dans ce chapitre, notre solution à la problématique posée au début de rapport. Nous avons obtenu de bons résultats. Cependant, certaines méthodes, utilisées dans les différentes étapes de la solution, présente des capacités assez limitées. Ces méthodes peuvent être subi à des amélioration et servir à améliorer les résultats.

Conclusion générale

Au cours de ce projet, notre contribution consiste en l'affectation, à la lampe rebotisée, la capacité de réagir (effectuer des comportement) en fonction des différents changements d'expressions des personnes à travers des expériences.

La solution que nous avons proposé comporte plusieurs étapes. Nous avons commencé par le prétraitement pour améliorer les résultats. Puis, la détection du vissage par le choix de la méthode basée sur l'histogramme de gradients orientés qui donne de meilleurs résultats comparé aux autres méthodes mais avec quelques limitations. Par la suite, nous avons utilisé un model basé sur les arbres de régressions pour extraire 68 points de saillances. Ce model est parmi les modèles les plus précis dans la littérature. Cependant, il présente un manque de précision pour certaines régions du visage très utiles pour notre application car cela est due à la base utilisée pour entrainer le model. Nous avons terminé notre méthode par l'utilisation d'une variante dynamique de la carte auto-organisatrice a deux dimensions. Cette dernière offre un grand avantage au niveau d'apprentissage en temps réel en termes d'espace et du temps.

Notre solution a montrée de très bonnes performances. Cependant, elle peut être sous réserve de plusieurs améliorations que nous avons envisagés de faire, mais malheureusement et par faute du temps, nous n'avons pas pu les mettre en place. Parmi ces perspectives nous citons :

- Entrainer un nouveau model pour l'extraction des points de saillances sur des données plus adaptées à notre application.
- Utiliser la régression pour encoder les changements d'expressions, et donc la possibilité de traiter plusieurs images de flux qu'on veut.
- Mettre en place des méthodes d'évaluation de performances de model du clustering et implémenter d'autres méthodes performantes afin d'en comparer.

Bibliographie

[1] Virginie André and Yann Boniface. Quelques considérations interactionnelles autour d'une expérience robotique. In *Workshop ACAI 2018*, 2018.

[2] Stanislas BARBILLON, Marjorie JULIO, and Aurélien STAB. Lampe robot, 2017.

[3] Charles Crouspeyre. Comment les réseaux de neurones à convolution fonctionnent, 2017.

[4] Eva D'Alessandro and Augustin Giovinazzo. Initiation à la recherche suivi de visages et détection de saillances, 2018.

[5] Samy Dany, Tram Hugo, Nogatchewsky Matthieu, and Bauer Simon. La pinokio lampe, 2016.

[6] NOWAK Julien and RIMLINGER Stéphane. Projet de découverte de la recherche suivi de visage et détection de saillances, 2018.

[7] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1867–1874, 2014.

[8] Teuvo Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9) :1464–1480, 1990.

[9] E. Lebarbier and T. Mary-Huard. Classification non supervisée. Master's thesis, AgroParisTech, 2016.

[10] Manuel Rebuschi. Psyphine.

[11] Buisine Roman and Riou Oksana. Suivi de visage et detection d'expression, 2018.

[12] Nicolas Rougier and Yann Boniface. Dynamic self-organising map. *Neurocomputing*, 74(11) :1840–1847, 2011.

Résumé

Dans le cadre du projet Psyphine qui vise à explorer et étudier les interactions entre l'homme et la machine ainsi, que l'attribution d'intentions et de conscience à cette dernière. Ce rapport présente notre contribution au projet, à savoir, la conception d'un système qui donne, à un prototype robotisé, la capacité d'interagir suivant les différents changements d'expression perçues. Le système comporte plusieurs phases dont la première s'agit de l'emploi de la méthode d'histogramme des gradients orientés combinée avec l'apprentissage automatique pour extraire des points de saillances qui seront par la suite utilisés dans le calcul des vecteurs caractéristiques des changements pour enfin effectuer un clustering utilisant la carte auto-organisatrice dynamique.

mots clés : Psyphine ; intéractions ; détection ; HOG ; dlib ; apprentissage-automatique ; carte auto-organisatrice dynamique

Abstract

As part of the Psyphine project, which aims to explore and study the interactions between man and machine as well as the attribution of intentions and consciousness to the latter. This report presents our contribution to the project, namely, the design of a system that gives a robotic prototype the ability to interact according to the different perceived changes in expression. The system consists of several phases, the first of which is the use of the oriented gradient histogram method combined with machine learning model to extract landmark points that will then be used to calculate the characteristic vectors of the changes and finally perform clustering using the dynamic self-organizing map.

Keywords : Psyphine ; interactions ; face-detection ; HOG ; machine-learning ; dlib ; DSOM