



RECONSTRUCTION DE LA FORME D'UN VISAGE À PARTIR DE PHOTOGRAPHIES

Rapport final PSC - INF 08

2020 - 2021

**MAXIME BONNIN PAUL LAUDRUP SICHENG MAO
YUTONG MENG VERONIKA SHILOVA**



TABLE DES MATIÈRES

1 Executive summary	4
2 État de l'art des Méthodes récentes de reconstruction faciales 3D	5
2.1 3D Morphable face Model	5
2.2 Méthode de reconstruction par apprentissage faiblement supervisé	6
2.3 Méthode basée sur les réseaux adverses génératifs GANs	7
2.4 Méthode de régression Volumétrique directe par des réseaux de neurones convolutifs	8
3 Base théorique et conceptuelle	10
3.1 Approche Machine Learning avec Scalismo	10
3.1.1 Scalismo : Qu'est-ce que c'est?	10
3.1.2 Pourquoi choisir Scalismo?	10
3.2 Probabilistic Morphable Models(PMMs)	11
3.2.1 Statistical Shape Modeling	12
3.2.2 Probabilistic Fitting	15
4 Démarches d'implémentation	20
4.1 Les raisons pour lesquelles nous avons choisi pytorch	20
4.2 Démarche de modélisation de tout le projet	21
4.3 Présentation des classes et de la classe modélisation	22
4.3.1 Diagramme schématique de construction	22
4.3.2 Facemodel	23

4.3.3	Individu	23
4.3.4	Render	24
4.3.5	Model	24
5	Résultats	26
5.1	Premiers résultats	26
5.1.1	Comparaison 80 paramètres contre 160 paramètres	26
5.1.2	Comparaison avec et sans fov	27
5.1.3	Mauvaise initialisation	28
5.2	Résultats Finaux	29
5.2.1	Résultat pour Donald Trump et Jennifer Lawrence avec fov	29
5.2.2	Modèle avec 3 photos en entrée	30
5.2.3	Problème convergence sur des visages Asiatiques et Afro-américain .	32
5.3	Améliorations possibles	33
5.3.1	Les contours	33
5.3.2	Qualité de la photo en entrée	34
6	Conclusion	35

1

EXECUTIVE SUMMARY

L'estimation de surface faciale 3D et autres composants intrinsèques du visage à partir d'une image unique est un problème très important à l'intersection de la vision par ordinateur et apprentissage automatique avec d'innombrables applications (par exemple, reconnaissance faciale, édition de visage, réalité virtuelle).

C'est pourquoi, les communautés d'infographie et de vision par ordinateur (computer vision) ont consacré des efforts de longue date à la création d'outils informatisés pour la reconstruction, le suivi et l'analyse de visages humains basés sur des entrées visuelles. Au cours des dernières années, des progrès rapides ont été réalisés, qui a conduit à des algorithmes nouveaux et puissants qui obtiennent des résultats impressionnantes même dans le cas très difficile de la reconstruction. La gamme d'applications est vaste et ne cesse de croître au fur et à mesure que ces technologies s'améliorent en terme de vitesse, précision et facilité d'utilisation.

Motivés par ces applications et la diversité de ce domaine, nous avons mis en place un algorithme utilisant une descente de gradient à partir d'une fonction de coût pour reconstruire un visage 3D. Nous avons d'abord créé un modèle simple qui reconstruit le visage à partir d'une photo. Nous l'avons amélioré au fur et à mesure des tests pour être plus robuste et reconstruire plus fidèlement la forme. Nous avons aussi tenté une approche de reconstruction à partir non pas d'une seule photo mais de trois photos d'une même personne.

2

ÉTAT DE L'ART DES MÉTHODES RÉCENTES DE RECONSTRUCTION FACIALES 3D

Les progrès rapides ont fait de ce domaine de l'informatique un domaine très actif en recherche, les méthodes de reconstruction de visage 3D à partir étant très importantes et ne pouvant pas à elle seule être toutes présentées dans ce rapport nous allons dans cette partie présenter donc les tendances récentes en matière de reconstitution faciale basées sur des technique d'apprentissage machine car il s'agit de la base de notre travail. La plupart de ces méthodes sont des algorithmes de reconstruction basés sur l'optimisation de fonction de coût. Nous fournissons un aperçu large des concepts sous-jacents à ces méthodes, et nous discutons d'hypothèses qui rendent ces algorithmes pratiques.

2.1 3D MORPHABLE FACE MODEL

Avant de présenter ces méthodes, il est indispensable de définir ce qu'est un "3D Morphable face Model" car c'est un concept à la base de plusieurs des tendances récentes en matière de reconstruction de la géométrie faciale.

Un "3D Morphable face Model" est un modèle génératif pour la forme et l'apparence du visage qui est basée sur deux idées clés : Premièrement, tous les visages peuvent être représentés comme des vecteurs. Un visage est alors un ensemble de N points de l'espace et en représentant les 3 coordonnées de chacun de ces N points dans un seul vecteur de taille $3N$, qui est généralement établi sur un ensemble d'exemples de visage grâce à une procédure d'enregistrement. En raison de cette représentation vectorielle, des combinaisons linéaires de visage peuvent être définies d'une manière significative, produisant des visages morphologiquement réalistes. La deuxième idée est de séparer la forme et la couleur du visage et de les distinguer des facteurs externes, c'est-à-dire les paramètres de lumière et de caméra (notions définies dans la section 3). Un "3D Morphable face Model" est basé sur des modèles statistique de la distribution des visages.

2.2 MÉTHODE DE RECONSTRUCTION PAR APPRENTISSAGE FAIBLEMENT SUPERVISÉ

L'objectif dans cette méthode est d'obtenir une reconstruction 3D précise du visage avec un apprentissage faiblement supervisé. Les auteurs du papier de recherche[20] ont remarqué que l'utilisation d'informations de bas niveau de couleur pixel par pixel uniquement et sans tenir compte des corrélations qui pouvaient exister entre les pixels vivant dans un voisinage proche peut poser des problèmes de minimum local où une faible erreur peut être obtenue avec des formes de visage insatisfaisantes. D'autre part, l'utilisation de la seule fonction de coût perceptive globale du visage conduit également à des résultats sous-optimaux car elle ignore la cohérence pixel par pixel avec l'image d'entrée. À la lumière de cela, cette approche propose une fonction de perte de niveau hybride qui intègre les deux, donnant lieu à des résultats précis. Cette approche propose également une nouvelle couleur de peaux à base de stratégie d'attention d'erreur photo-métrique, accordant à cette méthode plus de robustesse à l'occlusion et à d'autres variations d'apparence difficile telles que la barbe et le maquillage épais. Les principales contributions de cette méthodes sont :

- Elle propose une méthode de reconstruction de visage basée sur un réseau neuronal convolutif (Convolutional Neural Network ou CNN) qui exploite les informations sur l'image de manière hybride. La fonction de perte est constituée d'une fonction de coût perceptive qui prend en compte l'image de manière globale et d'une perte plus robuste qui prend en compte les détails de l'image. Le fait de combiner ces deux pertes rend la méthode plus performante que les autres méthodes complètement supervisées. En se basant sur le modèle de réseau de neurones permettant la reconstruction de la forme 3D à partir d'une seule image, cette approche permet aussi de considérer le problème de reconstruction à partir de plusieurs image car il est naturel de mettre à profit plusieurs images pour pouvoir avoir un meilleur modèle 3D. Pour atteindre cet objectif cette utilise une stratégie plus intelligente que la stratégie naïve qui consiste à agréger les résultats de reconstruction pour chaque image et faire une moyenne.
- Elle propose un nouveau mode d'apprentissage de la confiance des formes pour l'agrégation de reconstruction de visage multi-images. En effet elle utilise une réseau de neurone pour apprendre des mesures de confiance ou de qualité que l'on accorde au modèle 3D obtenu à partir de chaque image et les utilise pour obtenir la forme 3D final en faisant une opération qui s'apparente à une moyenne pondérée par ces mesures de qualité des différents modèles 3D obtenu pour chaque image.

2. ÉTAT DE L'ART DES MÉTHODES RÉCENTES DE RECONSTRUCTION FACIALES 3D

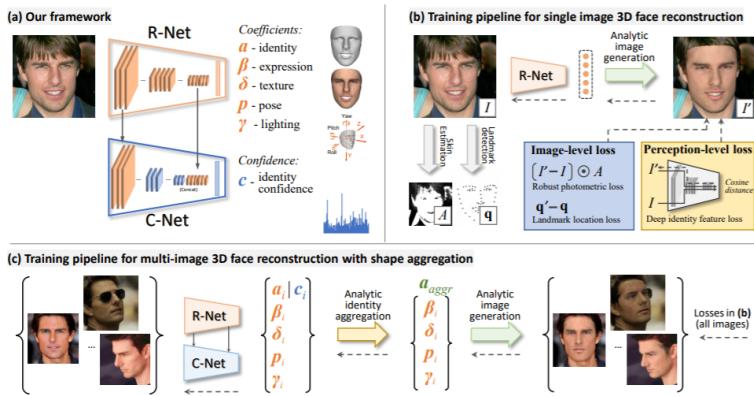


FIGURE 1 – Architecture et fonctionnement

2.3 MÉTHODE BASÉE SUR LES RÉSEAUX ADVERSES GÉNÉRATIFS GANs

Cette méthode[21] est elle aussi basée sur l'utilisation d'une architecture de réseaux de neurones profonds mais aborde de manière radicalement différente le procédé de reconstruction 3D de la forme et de la texture du visage en formulant une nouvelle stratégie de fitting du modèle 3D basée sur un réseau adverse génératif; une particularité de cette méthode[21] est qu'elle conçoit une nouvelle fonction de coût qui combine plusieurs pertes qui prennent en considération des caractéristiques d'identité profondes du visage.

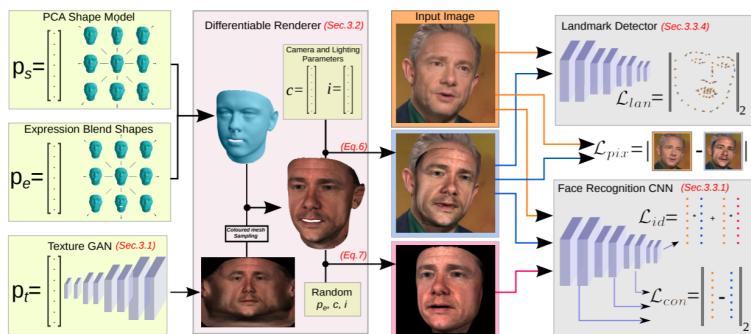


FIGURE 2 – Architecture et fonctionnement

Cette méthode[21] étend juste les méthodes de fitting 3D généralement utilisées en partant d'un modèle de forme 3D obtenu par des méthodes statistiques mais dont la texture est donnée par le réseau générateur GAN puis celui-ci est projeté en 2D par un moteur de rendu différentiable. Ensuite la distance entre le rendu 2D et l'image d'entrée est minimisée par le biais d'une fonction de coût qui prend en compte différents paramètres de forme et de texture. Cette approche a démontré d'excellents résultats au niveau de la reconstruction

de forme et de la texture du visage dans des conditions d'enregistrement arbitraires ; et les résultats obtenus se révèlent être très photo-réaliste.

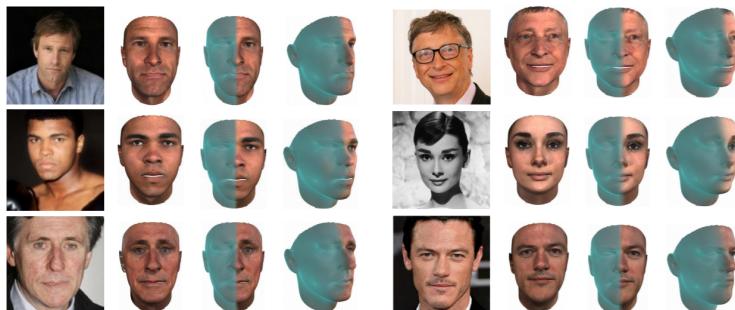


FIGURE 3 – Exemple de résultat obtenu

2.4 MÉTHODE DE RÉGRESSION VOLUMÉTRIQUE DIRECTE PAR DES RÉSEAUX DE NEURONES CONVOLUTIFS

C'est une approche qui contourne nombre des difficultés rencontrées dans la reconstruction de visage 3D en utilisant une nouvelle représentation volumétrique de la géométrie faciale et une architecture de réseau neuronal convolutif appropriée qui est formée pour régresser directement à partir d'une image faciale 2D à volume 3D correspondant. Un aperçu de cette méthode est illustrée à la Fig. 4. En résumé, les apports de cette approche sont :

1. Étant donné un jeu de données composé d'images 2D et de représentation 3D de visages, elle cherche si un réseau neuronal convolutif peut apprendre directement, de bout en bout, la correspondance entre les pixels de l'image et la représentation 3D complète de la structure du visage (y compris les parties du visage non visibles). En effet, les auteurs du papier de recherche[19] montrent que la réponse à cette question est positive.
2. Ils démontrent que leur réseau de neurone fonctionne avec une seule image 2D de visage et elle fonctionne de façon correcte sur des expressions faciales arbitraires, et peut être utilisée pour reconstruire toute la géométrie faciale 3D en contournant la construction (pendant l'entraînement) et l'ajustement (pendant le test) d'un 3DMM (pour 3D Morphable face Model).
3. ils y parvient grâce à une architecture neuronale convulsive simple qui effectue une régression directe d'une représentation géométrique 3D du visage à partir d'une seule image 2D. Le fitting d'un model 3DMM n'est pas utilisé. Cette méthode utilise uniquement des images 2D comme entrée de l'architecture CNN proposée.

Contrairement à la plupart des méthodes de reconstruction 3D, cette méthode est directe. elle n'évalue pas les paramètres d'un 3DMM et, en fait, elle contourne complètement l'ajustement d'un 3DMM. Au lieu de cela, cette méthode produit directement une représentation 3D du visage. En raison de cette différence fondamentale, cette méthode est également radicalement différente en termes d'architecture neuronale.

Le but de cette méthode est de prédire les coordonnées des différents sommets de la forme 3D du visage en se basant sur la correspondance avec l'image 2D et ceci par le biais d'une régression basée sur un réseau de neurone convolutif.

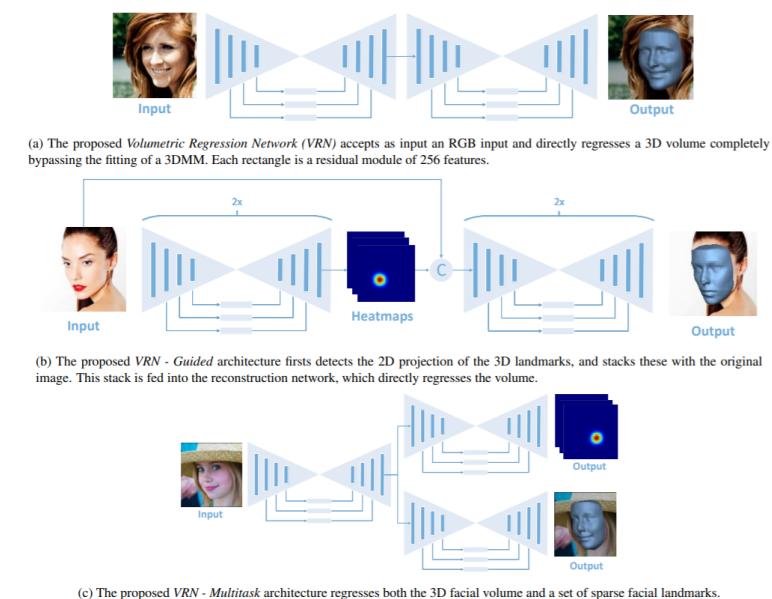


FIGURE 4 – Architecture d'un Réseau de Régression Volumétrique

3

BASE THÉORIQUE ET CONCEPTUELLE

Notre sujet appartient au domaine de Vision par Ordinateur et Traitement d’Image (Computer Vision and Image Processing en anglais). La théorie probabiliste et statistique constitue une composante majeure des bases théoriques de ce projet.

3.1 APPROCHE MACHINE LEARNING AVEC SCALISMO

3.1.1 • SCALISMO : QU’EST-CE QUE C’EST ?

Scalismo[14] est une bibliothèque de modélisation statistique de formes et d’analyse d’images basée sur des modèles codés en Scala[18], développée par le groupe de recherche sur les graphismes et la vision de l’Université de Bâle. Il est basé sur le concept des modèles probabilistes morphables (Probabilistic Morphable Models ou PMMs).

La vision de Scalismo est de fournir un environnement de modélisation et d’analyse d’images qui :

1. rend facile et amusant l’expérimentation d’idées et de construire des prototypes de recherche.
2. est suffisamment puissant pour créer des applications industrielles à grande échelle.
3. permet de le déployer dans des pipelines d’imagerie complexes et distribués.

Il y a deux concepts mathématiques majeurs qui forment la base conceptuelle de Scalismo :

1. Processus gaussiens pour modéliser les variations de forme, texture et des paramètres de rendu.
2. Markov Chain Monte Carlo method (MCMC method) pour l’ajustement du modèle.

3.1.2 • POURQUOI CHOISIR SCALISMO ?

La raison de ce choix est simple :

1. Il est open source, libre d’utilisation et de distribution.

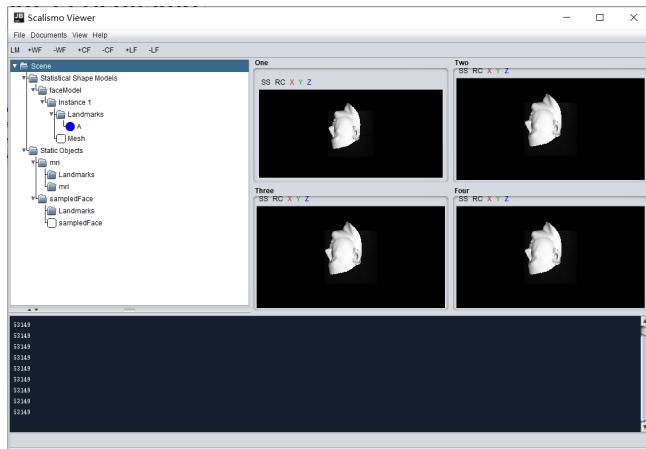


FIGURE 5 – Utilisation de Scalismo GUI

2. Il fournit des données réalistes et des tutoriels pour les débutants.
 3. Basé du langage Scala, il hérite tous les avantages de Scala.
 4. Issu d'un cadre médical visant à analyser des images anatomiques, il correspond bien à notre objectif : l'analyse des visages humains.

3.2 PROBABILISTIC MORPHABLE MODELS(PMMs)

Les PMMs sont utilisés pour une analyse d'image basée sur un modèle utilisant une approche d'« analyse par synthèse ». Le cadre se divise naturellement en une composante pour la modélisation statistique d'objets et une composante pour l'adaptation d'un tel modèle à de nouvelles données. L'analyse de nouvelles données est effectuée en adaptant des modèles d'objets statistiques aux données à l'aide de l'optimisation Markov Chain Monte Carlo (MCMC). La figure 6 illustre les structures de PMMs.

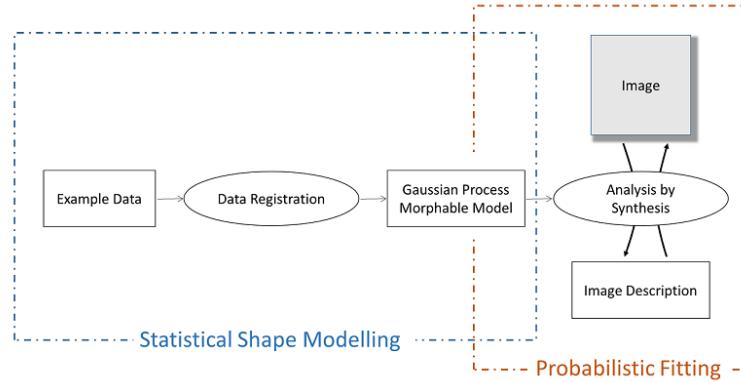


FIGURE 6 – Structure de PMMs

3.2.1 • STATISTICAL SHAPE MODELING

Dans le Statistical Shape Modeling, il y a des notions de bases importantes pour notre projet.

0. Modèle Statistique : Un modèle statistique dans notre cadre est une famille de lois gaussiennes modélisant l'ensemble des visages possibles. Par exemple, le Basel Face Model qu'on utilise dans notre étude est un modèle qui code les informations de la forme ainsi que la texture des visages. Voir la section 2 ci-dessous pour plus de détails.

1. Forme et Landmarks : La forme est la propriété centrale de l'objet qu'on étudie. Dans le Statistical Shape Modeling, la forme est représentée par un ensemble de points. Parmi les points de cette forme, il peut y avoir d'autres points, bien importants qui sont des points caractéristiques de la forme. Ils ont souvent une signification anatomique, par exemple, le bout de pouce, le bout de nez.

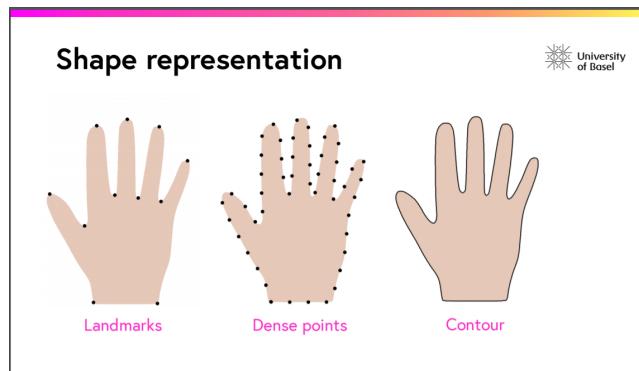


FIGURE 7 – Forme et landmarks

Dans notre cas, les landmarks du visage sont les points clés du visage. Ils indiquent la position des yeux, du nez, de la bouche et du contour du visage. La figure 8 montre les

résultats que l'on a obtenu pour la détermination des landmarks avec la bibliothèque dlib[5].



FIGURE 8 – Landmarks du visage

2. Loi Gaussienne et Modèle de forme : La loi Gaussienne domine une grande partie de phénomènes aléatoires. Par exemple dans notre étude, tous les visages humains différents forment une famille. Cette famille suit une loi gaussienne par rapport aux paramètres de forme. Le choix de la paramétrisation du visage est discuté dans la partie d'Analyse en composantes principales. Une telle Gaussienne s'appelle aussi un modèle de forme. Une gaussienne se caractérise par sa moyenne et sa variance. La variance selon une composante indique son degré d'influence sur la forme d'un visage.

Il est aussi important de noter que ces paramètres sont extraits à partir de l'image par la machine et ne sont donc généralement pas compréhensibles du point de vue de l'humain (par exemple, la taille des yeux ou la hauteur du nez) car les paramètres trouvés sont en général différents de ce que l'on peut s'imaginer comme la taille des yeux, la hauteur du nez, la largeur,... Il peut arriver que des paramètres aient un sens physique compréhensible mais cela n'est pas toujours le cas. Pour l'illustrer, on peut citer une [vidéo](#) qui se rapproche de notre sujet dans laquelle la signification des composantes du PCA est décortiquée.

La loi de Gaussienne nous permet non seulement de classifier les visages données mais aussi de générer un nouveau visage à en modifiant les paramètres.

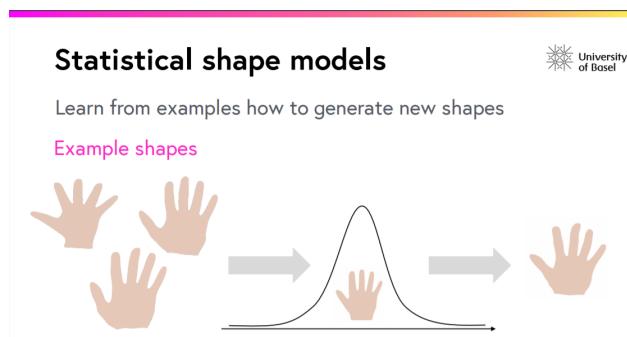


FIGURE 9 – Loi gaussienne

3. Déformation et Mesh : Une déformation est un champ de vecteur dont le vecteur pointe du point de la forme source au point correspondant de la forme cible, qui aussi suit une loi gaussienne. Dans le cas discret, il peut s'exprimer par une loi gaussienne multivariante, tandis que dans le cas continu, il se peut exprimer par une processus gaussien. La construction d'un processus gaussien à partir d'une loi gaussienne multivariante et la discréétisation d'un processus gaussien en une loi gaussienne multivariante dépendent d'une propriété de la loi gaussienne appelée marginalité. En pratique, il faut toujours discréétiser le modèle continu et dans notre cas on discréétise le visage en une forme à partir d'un nombre fini de point comme le montre la figure 10.

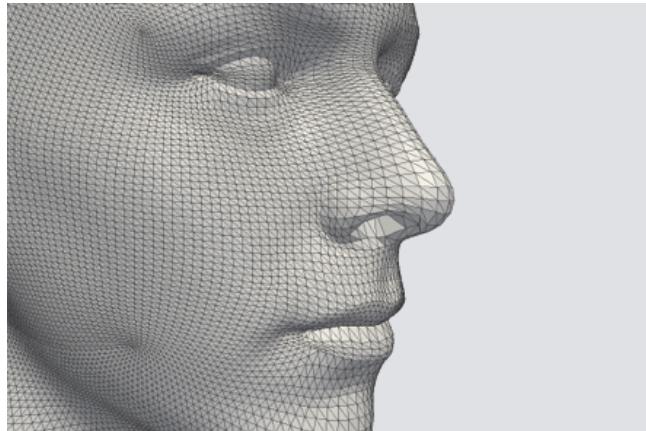


FIGURE 10 – Mesh

Maintenant, on est prêt à introduire les méthodes principales que l'on utilise dans notre projet.

4. Analyse en composantes principales, PCA : Comme la figure 10 nous le montre, la discréétisation est très fine car la forme est construite avec environ 58 000 points et 160 000 triangles. Une loi gaussienne d'une telle dimension n'est pas faisable en pratique à cause de la limite du calcul et de la mémoire. En effet, il faudrait calculer une matrice de covariance de 58 000 par 58 000. Pour simplifier le problème en gardant les informations utiles, on applique une méthode qui s'appelle l'"analyse en composantes principales" (Principal Component Analysis). Grâce au théorème de Karhunen–Loève, un processus gaussien peut se décomposer en une série de lois gaussiennes de variances différentes. Si on trie ces lois selon leurs variances et garde celles de variances les plus grandes, on peut également obtenir une approximation suffisamment précise par rapport au processus d'origine, ce qu'on appelle une approximation de rang fini. La tâche de PCA est de trouver les composantes avec la variance la plus grande dans cette décomposition. Attention, l'approximation de rang fini n'est pas toujours efficace, surtout lorsque la variance de composante ne décroît pas rapidement, mais ce n'est pas le cas dans notre étude (Voir la figure 11).

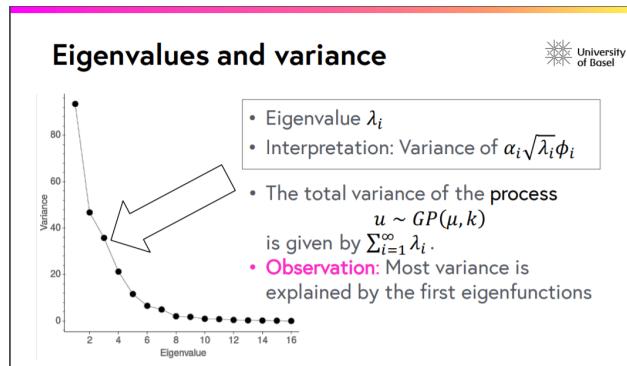


FIGURE 11 – les composantes avec ses variances

3.2.2 • PROBABILISTIC FITTING

Une fois qu'on construit le modèle de forme par rapport à une base de données, on peut l'utiliser pour étudier les formes à nouveau. La problématique du fitting probabiliste est, en bref, comment ajouter les formes qu'on veut étudier à notre modèle déjà construit. Le point clé est de trouver la correspondance entre la forme moyenne et la forme cible.

5. Inférence bayésienne : L'inférence bayésienne est une méthode centrale dans l'apprentissage de machine. Il s'agit d'un schéma de l'apprentissage :

1. avoir une connaissance a priori,
2. observer les évidences,
3. renouveler la connaissance a posteriori.

Exprimé par une formule :

$$\text{posteriori} = \frac{\text{vraisemblance} \times \text{priori}}{\text{vraisemblance marginale}}$$

Cette formule se base sur le théorème de Bayes dans la statistique bayésienne.

Mais en pratique, ce n'est pas facile à appliquer l'inférence bayésienne directement. C'est une difficulté qui se pose souvent dans les scénarios d'application pratique des statistiques. Cette difficulté est une contradiction entre la théorie et la pratique. Tout comme nous avons besoin avant, dans la pratique, de discréteriser des processus gaussiens continus en théorie en lois gaussiennes multivariées, la formule bayésienne est en un sens a priori, puisqu'il implique par défaut que l'on possède toutes les informations contenues dans le membre de droite dans cette formule, y compris la densité de loi. En pratique, cependant, nous ne pouvons obtenir qu'un nombre fini d'échantillons de cette loi, et nous devons estimer ou modéliser cette information préalable de la théorie de probabilité dans la pratique statistique à l'aide de ces seuls échantillons finis.

L'idée principale est de simuler la loi et de remplacer toutes les intégrations par des sommes empiriques évaluées sur des représentants simulés de la loi, appelés échantillons aléatoires. Parmi plusieurs algorithmes réalisant cette idée (ceux qu'on appelle les méthodes de Monte-Carlo par chaînes de Markov, MCMC), nous choisissons l'algorithme de Metropolis-Hastings.

6. Algorithme de Metropolis-Hastings : Cette algorithme provient d'une heuristique très intuitive.

1. Proposer un échantillon d'une loi simple (à manipuler pour nous)
2. Vérifier s'il est satisfaisant par rapport aux critères liés à la loi cible.

En réalité, on applique cette idée d'une façon itérative pour mieux modéliser. L'algorithme complet est présenté plus loin.

Retenons qu'on veut mettre la correspondance entre un modèle 3D et une 2D image cible. Comme on a dit avant, ce processus suit un schéma d'analyse par synthèse.

7. Analyse par synthèse : Tout d'abord, présentons le modèle de visage nommé Basel Face Model ou BFM [2]. C'est un modèle de visage paramétrique et génératif accessible publiquement [1]. Il s'agit d'un modèle statistique construit à partir de 100 scans 3D de haute qualité de femmes et de 100 hommes. Nous pouvons l'appliquer comme un a priori pour la forme et la couleur afin de synthétiser des visages 3D. Ce modèle comporte de quoi construire la structure 3D du visage "moyen", la texture moyenne du visage mais aussi les composantes principales pour la forme et la texture.

Si nous sommes capables de générer une image synthétique proche d'une image cible observée à partir d'un modèle significatif, alors nous aurons une description de l'image - c'est l'idée essentielle de l'analyse par synthèse. Dans la suite, on adapte cette idée au cadre d'inférence bayésienne dont il y a cinq défis principales :

1. On a besoin d'un modèle de visage paramétrique génératif.
2. On a besoin d'un moyen de générer des images synthétiques.
3. Comment générer une image a priori par rapport au modèle de visage paramétrique génératif ?
4. Comment définir une vraisemblance pour mesurer la similarité entre une image synthétique et l'image cible ?
5. Comment trouver ou inférer des solutions appropriées respectivement pour générer des images synthétiques proches de l'image cible ?

Pour 1, on utilise la modélisation du visage dans le BFM. Pour 2 et 3, on utilise la technique qui s'appelle rendu dont on parlera par la suite. Pour 4, on utilise la distance de moyenne quadratique entre l'image du rendu et l'image cible, mais il convient de noter que la contribution des paramètres différentes à cette erreur n'est pas de la même grandeur, ce qui résulte une coordination attentive dans notre exécution de l'algorithme d'inférence bayésienne. À propos de 5, d'une part on a besoin de l'algorithme de Metropolis-Hastings qui générer les

Filtering: Multiple MH Decisions

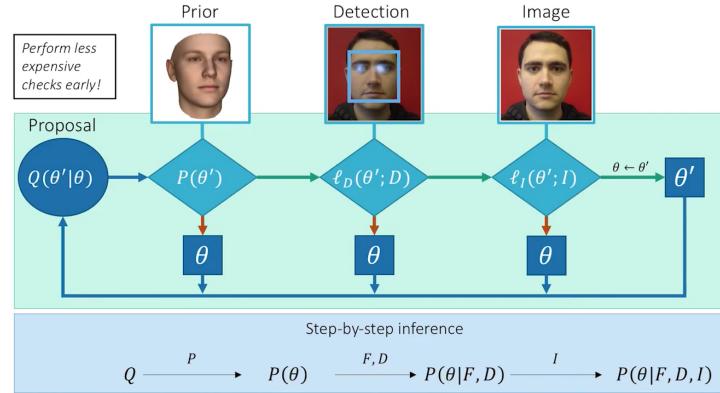


FIGURE 12 – Le schéma de fitting

échantillons et d'autre part, on a besoin d'une méthodologie pour traiter le problème de simulation dans un ordre logique(Voir aussi la figure 12 :

1. Générer une image a priori
2. Déetecter et adapter la région de visage (la position, la taille)
3. DéTECTER et adapter les landmarks de visage
4. Adapter l'image entière.

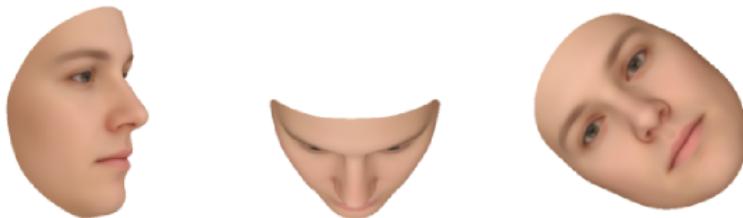


FIGURE 13 – Caméra



FIGURE 14 – Lumière

Pour mesurer la similarité d'une visage générée par rapport à l'image cible, il faut que l'on choisisse les paramètres significatifs d'image et en déduire une image à partir du visage 3D qui reflète ces paramètres et qui est donc comparable avec la cible. C'est ce que fait la partie de rendu.

8. Rendu à partir de Forme, Caméra, Lumière et Couleur : Pour faire les correspondances entre un modèle 3D et une image 2D, c'est-à-dire comparer ce qu'on produit avec ce qu'on vise, on fait un rendu de notre modèle 3D. Précisément, à partir des coordonnées 3D des points sur une surface générée par le modèle, qui codent toutes les informations sur la forme, prenant en compte également des autres paramètres liés à l'image : la pose de caméra (Figure 13), l'éclairage (Figure 14) et la couleur, on calcule les pixels dans la grille correspondante. Le processus entier, ressemble à la prise d'une photo mais par simulation.



FIGURE 15 – Image cible et le rendu de Scalismo

Une génération de rendu se fait en trois étapes :

1. calcul de l'éclairage (on parle d'illumination) : Cela comprend l'éclairage local (illumination face à face, ombrage) et l'éclairage global qui calcule les échanges d'énergie lumineuse entre les éléments en tenant compte de l'absorption et de la réflexion.
2. projection dans le plan d'observation : Cela s'appuie sur une "caméra" qui nous per-

met de projeter les points sur un écran. Cette caméra peut être perspective ou orthographique. La première donne une impression de point de fuite des objets, tandis que la seconde conserve la taille des objets, comme si nous les regardions de loin. La caméra de perspective correspond à la façon dont on voit les choses.

3. dessin à proprement parler avec application éventuelle d'une texture.

9. Algorithme final : Maintenant avec tous ces outils à la main, on peut construire un algorithme complet :

Algorithm 1: Génération de visage 3D à partir d'une image cible

```

Entrée : Image d'un visage
Sortie : Visage 3D
On part de  $I$ , l'image de départ
On part du visage moyen  $X_0$ 
// Tout d'abord, traiter la caméra
Tirer une rotation  $R$ 
while le nombre d'itération est inférieur à  $N$  do
    Créer  $X_{n+1}$  à partir de  $X_n$  grâce à  $R$ 
    Calcul de la perte entre  $I$  et  $I_n$ 
    if la perte est plus faible then
        Garder  $X_{n+1}$ 
    else
        Garder  $X_{n+1}$  avec une certaine probabilité
    end if
end while
Faire de même pour la taille du visage
Faire de même pour la forme avec un tirage gaussien de la déformation selon les composantes principales
Faire la même chose pour la couleur que pour la forme
  
```

Remarque :

1. L'image I_n est un rendu du modèle X_n .
2. La boucle est l'algorithme de Metropolis-Hastings.
3. L'algorithme entier est une inférence bayésienne.

4

DÉMARCHES D'IMPLÉMENTATION

4.1 LES RAISONS POUR LESQUELLES NOUS AVONS CHOISI PYTORCH

Nous avons commencé par expérimenter Scalismo[14], qui était écrit en Scala[18] mais cela n'était qu'un **point intermédiaire** de notre projet. Il nous a permis de comprendre ce que l'on allait manipuler : le modèle BFM[2] et les notions théoriques relatives à la reconstruction 3D. Nous avons décidé de quitter Scalismo car le langage Scala ne nous était pas familier et surtout on ne voyait pas comment on allait apporter notre pierre à l'édifice. Scalismo nous apparaissait comme une boite noire car nous n'avons pas accès au code source. Pour pouvoir innover et expérimenter, la manière la plus simple est donc de faire le plus de choses nous même pour avoir une compréhension la plus complète de ce qu'il se passe.

Nos tuteurs nous ont présenté Pytorch3D[13] une librairie qui étend Pytorch[12] à la 3D. On utilise Pytorch et Pytorch3D car elles permettent une descente de gradient simplifiée. Celle-ci concerne la forme de la structure 3D, le « renderer » (objet de rendu 3D) de la structure 3D et la texture. Dans notre problème une descente de gradient est bien plus efficace que les algorithmes de type Metropolis-Hasting. Ils sont certes très généraux mais le temps de convergence est grand et le résultat dépend beaucoup de l'état initial. En effet, la descente de gradient dépend de l'erreur et non pas du nombre d'itérations effectuées donc l'algorithme se bloque moins souvent dans des minima locaux que celui de Scalismo si les conditions initiales ne sont pas suffisamment proches du visage cible. En optimisant une fonction de coût caractéristique du problème, on peut éviter d'autant plus ce problème.

L'avantage de Pytorch et Pytorch3D est que nous pouvons choisir de calculer sur GPU ou CPU. Or le calcul parallèle est adapté à la génération de rendu et donc on peut en obtenir plus rapidement qu'en calculant sur CPU. Pour donner un ordre de grandeur, la génération de rendu sur CPU avec un ordinateur lambda prend jusqu'à une minute. Lorsque que l'on a besoin d'au moins 1000 ou 2000 itérations pour converger vers une solution descente, il paraît inconcevable de procéder ainsi. Avec un calcul sur GPU, on arrive à créer un rendu en moins d'une seconde.

4.2 DÉMARCHE DE MODÉLISATION DE TOUT LE PROJET

Pour notre approche on utilise le modèle de la tête [1] que l'université de Bâle utilisait sur Scalismo. En extrayant ces données, on peut construire la structure 3D de notre visage et créer un rendu. À gauche de la figure 16 représente le visage moyen de notre base de donnée. Dans cette base de données, ce que l'on appelle «**PCA_basis**» est donc un tableau dont les colonnes sont les vecteurs directeurs des composantes principales. Le tableau «**PCA_Variance**» correspond au tableau des variances des composantes principales du PCA. En clair, cette base de donnée nous fournit de quoi créer la structure du visage moyen et les outils pour la modifier. Avec cela on peut générer un visage de forme aléatoire avec une équation simple :

$$\text{Visage} = \text{VisageMoyen} + \text{SamplePCA}.\text{PCA}_{basis}$$

On peut faire la même chose pour la couleur avec la texture moyenne, le PCA de la couleur et la variance selon ce PCA. À droite de la figure 16, on montre un rendu de génération de forme



FIGURE 16 – A gauche le visage moyen et à droite un sampling de forme

Ces relations sont fondamentales car par la suite on remplace SamplePCA par des paramètres que l'on va optimiser. Utiliser des paramètres après un PCA permet de limiter le nombre de paramètres à utiliser et donc minimiser les calculs tout en conservant la qualité de la solution.

On sait désormais comme le visage sera modifié, on cherche donc comment on va pouvoir le faire converger vers notre image :

- **Landmarks** : Après des recherches sur le sujet, il s'est avéré que les landmarks étaient très utilisés. Leur rôle est de forcer la convergence en position de notre objet grâce à des points importants du problème. Dans notre cas, cela permet de rapprocher la position du visage de notre modèle sur celui sur la photo. Or si la position entre les deux visages est proche, notre solution sera plus à même de converger vers une solution satisfaisante, d'où l'intérêt d'utiliser les landmarks pour forcer la convergence en position du visage.
- **Couleur** : On doit aussi faire converger la texture car celle-ci contient des informations cruciales pour la forme. La façon de procéder qui a été choisie est de faire converger la rendu de notre modèle vers le visage sur la photo. Il faut ainsi considérer uniquement les pixels qui appartiennent au visage sur la photo.

Ainsi, les fonctions de coût caractéristiques de notre problème seront les suivantes :

$$\begin{aligned} \cdot L_{land} &= \frac{1}{68} \sum_{i=1}^{68} \|land_i - land_{proj,i}\|^2 \\ \cdot L_{Color} &= \frac{1}{Card(FacePixels)} \cdot \|Image_{Contour} - Image_{Render}\|^2 \\ \cdot L_{regShape} &= \sum_{i=1}^{160} \left(\frac{ParamShape_i}{\sigma_{Shape,i}} \right)^2 \\ \cdot L_{regColor} &= \sum_{i=1}^{160} \left(\frac{ParamColor_i}{\sigma_{Color,i}} \right)^2 \end{aligned}$$

Avec cette idée en tête, on a compris que le projet se découvrait en 3 parties pour répondre au problème :

- Modélisation du visage
- Analyse de l'image en entrée
- Outils de rendu 3D

Ces trois parties sont les éléments essentiels pour répondre à notre problème.

4.3 PRÉSENTATION DES CLASSES ET DE LA CLASSE MODÉLISATION

4.3.1 • DIAGRAMME SCHÉMATIQUE DE CONSTRUCTION

Pour comprendre la construction de l'algorithme qui permet de répondre au problème, on pourra se référer à la figure 17. Elle permet d'avoir une vision globale de l'architecture et des divers attributs ou fonctions importantes. Les paramètres à optimiser sont en rouge, les méthodes de classe en marron et en noir les attributs qui stockent des informations cruciales.

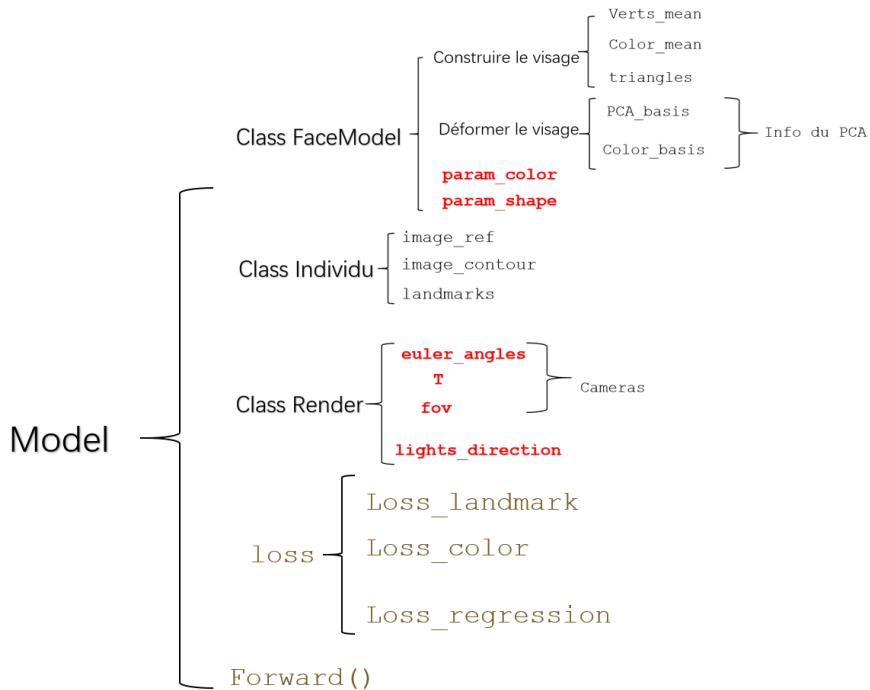


FIGURE 17 – Les liens entre les classes

4.3.2 • FACEMODEL

Dans cette classe, on charge les données de la base de données pour pouvoir construire tous les éléments de la structure 3D de type « Mesh ». Les deux attributs param_shape et param_color sont respectivement les paramètres de notre visage pour la forme et la couleur.

On connaît la variance pour chacune des composantes principales du PCA de la couleur et de la forme. On modélise chacune des variables du PCA par une variable gaussienne de moyenne 0 et de variance donnée par PCA_Var ou Color_Var. Les variables sont donc des variables de déformation du visage moyen. La méthode sample_face() réalise un tirage selon ce principe. La méthode mesh_mean() ne fait juste que créer un objet Mesh correspondant au visage moyen.

4.3.3 • INDIVIDU

Cette classe sert à calculer les 68 landmarks de notre image d'entrée avec la librairie dlib[4]. Puis elle nous permet de stocker les landmarks, la photo de référence dans un seul objet et la photo avec l'arrière plan retiré. Et si on ne donne pas d'image détournée, grâce à la librairie OpenCV[10] on supprime le fond.

4.3.4 • RENDER

Attributs

Dans cette classe, on crée tous les objets nécessaires pour créer un rendu de notre structure 3D. Certains sont des paramètres que l'on va chercher à optimiser. Les paramètres à optimiser sont les angles d'euler, la matrice de translation, la direction de la lumière et le field of view. Les angles d'euler et la matrice de translation permettent de déterminer la position de la caméra.

Méthodes

Render(mesh) : permet de créer un rendu de la structure "mesh" de type Mesh à partir des différents attributs de rendu (caméras, lumière,...) de l'objet sur lequel cette méthode est appelée.

compute_R() : permet de calculer la matrice de rotation à partir des angles d'euler.

4.3.5 • MODEL

Attributs

On utilise les classes définies précédemment pour créer trois objets Individu et trois objets Render à partir de trois photos en entrée et chaque paire (Individu, Render) est indépendante l'une de l'autre, en revanche ils partagent le même objet FaceModel.

Méthodes

Compute_projected_lmks(mesh),(get_points) : permettent de retourner la projection des landmarks de notre structure 3D. On connaît l'indice des sommets de notre structure Mesh qui correspondent aux 68 landmarks de la bibliothèque dlib. En connaissant cela on a juste à projeter tous ces sommets dans le plan de la caméra.

loss_land(), loss_reg(), loss_color() : permettent de calculer les pertes significatives du problème. La première est la distance entre les landmarks de notre image et ceux projetés. La deuxième est une perte de régression des paramètres de forme et de couleur pour éviter qu'ils n'explosent. La dernière est la norme deux entre notre visage projeté et le visage extrait de la photo (ce sont les pertes décrites en section précédente).

loss() : est une combinaison linéaire des 3 erreurs précédentes. La relation est proche de celle utilisée dans le papier de recherche[11]. On a juste diminué le poids de régression de la forme pour que la forme soit encore plus modifiée. :

$$L_{tot}[Index] = \lambda_{Color} \times L_{Color} + \lambda_{land} \times L_{land} + \lambda_{regShape} \times L_{regShape} + \lambda_{regColor} \times L_{regColor}$$

où les poids liés aux pertes de couleur, landmarks et régression sont : $\lambda_{Color} = 100$, $\lambda_{Land} = 1$, $\lambda_{regColor} = 2,5 \cdot 10^{-5}$, $\lambda_{regShape} = 2,5 \cdot 10^{-7}$

Et pour la version 3 caméras, on calcule séparément les pertes loss_land et loss_color, et loss_reg pour chaque paire (Individu, Render). Enfin on calcule la perte globale de la manière suivante :

$$Loss = \sum_{index=0}^2 w_i \times L_{tot}[index]$$

avec w_i : le poids pour l'individu i.

render_visual(), render() : renvoie le résultat du rendu du visage paramétré grâce à son attribut de type Rendu et affiche les résultats.

forward_individual() : Modifie les structures et les attributs autres que les paramètres pour une paire (Individu, Render) et renvoie la perte concernée.

forward() : En utilisant la fonction précédente, modifie les structures et les attributs autres que les paramètres et renvoie la perte totale.

optimization_individual(model, optimizer, scheduler, index, step, step_visu) : Elle permet d'optimiser notre modèle sur une seule paire (Individus, Render) en ne calculant seulement la perte totale de la paire considérée.

optimization(model, optimizer, scheduler, step, step_visu) : Cette fois, on optimise notre modèle selon la perte globale qui est une combinaison linéaire des pertes totales des trois paires.

5

RÉSULTATS

Dans cette section, nous présenterons la démarche scientifique que l'on a suivi une fois que nous avions une implémentation simple qui fonctionne. A travers des photos de nos résultats, nous montrerons ce qui a marché, n'a pas marché et pourquoi. Les différents points abordés seront les suivants :

1. Nombre de paramètres
2. Avec ou sans le paramètre de "field of view"
3. Cas d'une mauvaise initialisation
4. Comparaison entre une photo et trois photos en entrée
5. Les problèmes de convergence sur certains visages
6. Les possibles améliorations de notre implémentation

5.1 PREMIERS RÉSULTATS

5.1.1 • COMPARAISON 80 PARAMÈTRES CONTRE 160 PARAMÈTRES

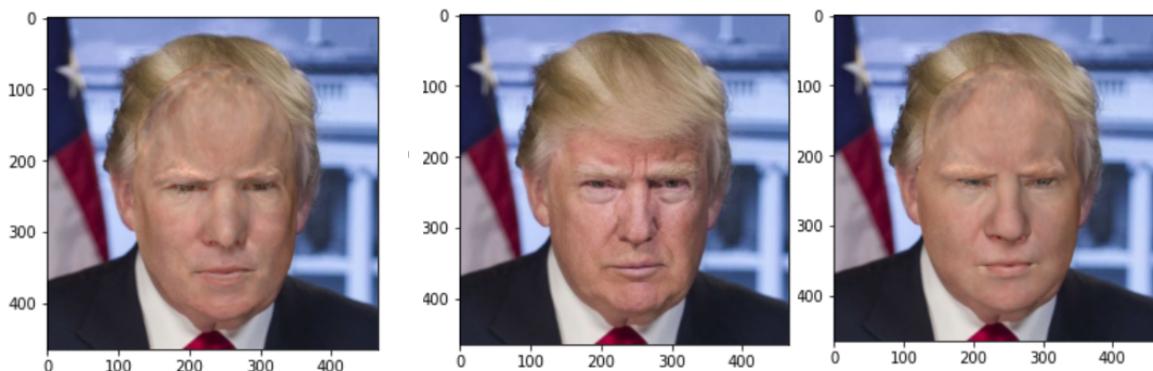


FIGURE 18 – Donald Trump : Rendu avec 80 paramètres (gauche), Image de référence (Milieu) et Rendu avec 160 paramètres (droite)

Sur la figure 18, on voit que les deux résultats sont correctes, en comparant avec la photo originale on reconnaît Donald Trump. Le résultat prend en compte l'expression faciale

5. RÉSULTATS

de Donald Trump comme en témoigne les sourcils. Cependant, en regardant la forme d'un point de vue local, on peut dire qu'avec 160 la forme est mieux reconstruite. Le nez et ce qu'il y a à côté se rapprochent plus de la forme réelle. De même pour la bouche, elle est plus large et similaire à celle sur la photo. Au niveau des yeux, avec 80 paramètres les yeux semblent disparaître, tandis qu'avec 160 cela n'est pas le cas. En clair, 80 paramètres sont suffisants pour obtenir un visage reconnaissable mais 160 paramètres permettent une amélioration de la forme.

De plus cette amélioration a un coût de calcul insignifiant par rapport au reste. Le génération de rendu est dominant dans la complexité temporelle. C'est pourquoi par la suite, nous conserveront les 160 paramètres pour toutes nos modélisations.

Néanmoins, on remarque un problème au niveau des oreilles et du bord car elles n'apparaissent pas.

5.1.2 • COMPARAISON AVEC ET SANS FOV

Sur les exemples précédents, Trump est très bien reconstruit. On peut donc juger que ces résultats sont satisfaisants. Néanmoins, en regardant en détail on remarque que les oreilles disparaissent de l'image. Ce problème a été dérangeant pour de nombreux tests, comme par exemple la photo de Nicolas Cage ci-dessous. Nous arrivons à reconnaître la personne lorsque nous superposons l'image produite avec l'image de référence. En regardant avec attention, on remarque que les pommettes, les zones proches des yeux, la bouche et du nez sont semblables à l'image. Cependant le rendu sans superposition nous montre que le résultat est loin d'être un visage humain et encore moins un visage qui ressemble à celui de Nicolas Cage.

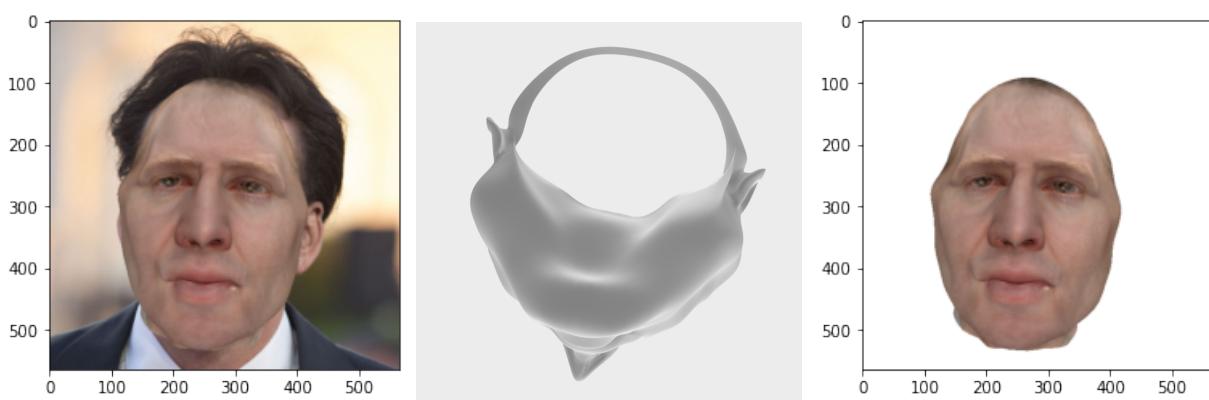


FIGURE 19 – Nicolas Cage : Résultat sans FOV

La solution fut d'utiliser le paramètre "field of view" de la caméra, il s'agissait simplement d'un problème de perspective. Notre solution marchait donc dans des conditions

5. RÉSULTATS

bien spécifiques et non pas dans le cas général. Avec ce nouveau paramètre que l'algorithme optimise nous pouvons converger vers un modèle 3D représentant d'avantage un visage humain. Avec ce paramètre et la même photo en entrée, on obtient cette fois ci les résultats suivants 20. Le résultat ainsi produit est largement satisfaisant si on oublie la texture proche des cheveux car notre algorithme n'est pas fait pour.



FIGURE 20 – Nicolas Cage : Comparaison entre référence et résultat avec FOV

5.1.3 • MAUVAISE INITIALISATION

Notre démarche d'optimisation au départ était d'optimiser tous les paramètres directement. Mais cela n'est pas idéal car si l'initialisation est mauvaise, notre modèle peut être déformé de manière irréversible lors de l'optimisation. Une idée a donc été de d'abord faire converger les paramètres de caméras avant de faire converger tous les paramètres en même temps. Sur la photo ci-dessous 21, on peut voir le résultat de convergence après 3000 itérations lorsque l'état initial est loin de la position initiale. Or, pour un état initial correct, en 3000 itérations le visage devrait être reconnaissable.

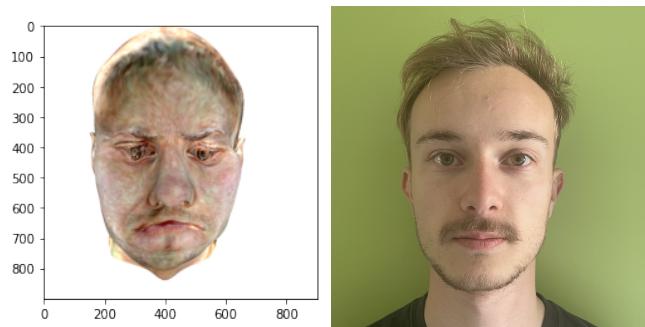


FIGURE 21 – Résultat de convergence après 3000 itérations pour une mauvaise initialisation

5.2 RÉSULTATS FINAUX

5.2.1 • RÉSULTAT POUR DONALD TRUMP ET JENNIFER LAWRENCE AVEC FOV



FIGURE 22 – Donald Trump : L'image initiale (à gauche) et l'image superposée (à droite)

En comparant avec les premiers résultats sans le paramètre de champ de vision, le rendu a des joues plus pleines mais la forme n'a pas beaucoup été modifiée. Mais il s'agit d'un cas particulier car la photo réelle est prise de loin. Donc le champ de vision n'a pas une grande différence dans ce cas. Un point positif est que désormais, on observe les oreilles et on a pas de sur-croissance des joues comme dans le cas de Nicolas Cage. Ainsi la forme dans n'importe quelle condition de rendu reste humaine.

On remarque que un coin du cou, la texture du visage est blanche à cause du col blanc que Trump porte. Et la coiffure de Trump pose aussi un problème sur le résultat du front car notre algorithme tente de faire converger la texture aussi à cet endroit.

L'exemple de la figure 23, montre que notre modèle se généralise pour les femmes. Ce qui était prévisible car la base de données est faite à partir d'autant de visage d'hommes que de femmes. Le nez est très bien reconstruit avec une forme pertinente, les sourcils et la bouche aussi. On a un très bon contour de visage correspondant au visage initial. Cependant la texture est mal reconstruite à cause du nombre de sources lumineuses. Notre modèle contient un modèle simple de lumière unique venant de l'infini. C'est pourquoi la texture est à «pois». Néanmoins, cela n'a pas empêché la forme de converger vers une forme cohérente avec la photo de référence.

5. RÉSULTATS

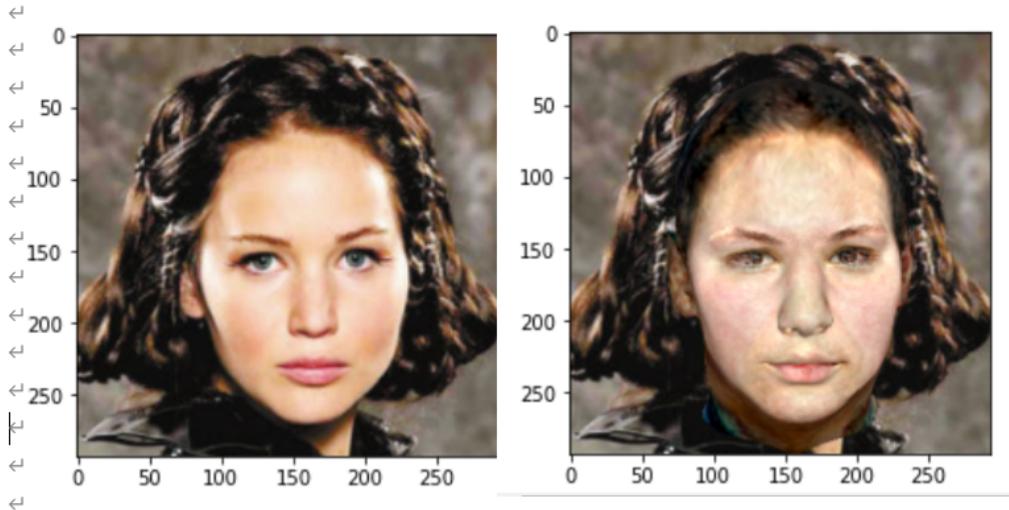


FIGURE 23 – Jennifer Lawrence : L'image initiale(gauche) et l'image superposée(droite)

5.2.2 • MODÈLE AVEC 3 PHOTOS EN ENTRÉE

Au début de notre recherche sur l'état de l'art, nous étions tombés sur une approche[9] qui combinait les informations de plusieurs images pour améliorer la qualité de la reconstruction. On a donc décidé de faire de même avec notre modèle en fournissant 3 photos d'une même personne, dans des conditions différentes (lumière ou angles de vue différents). Nous voulions améliorer certaines zones qui n'étaient pas parfaitement reconstruites avec une seule image comme le nez ou les arcades. L'image en figure24 représente la forme du nez lorsque l'on utilise une caméra uniquement. Dans de nombreux cas, le nez prend cette forme pointue. Cela nous a incité à nous lancer dans cette solution. Dans notre modèle on considère alors le même visage mais avec dans des conditions de rendu différentes.

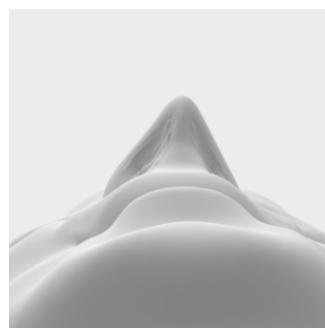


FIGURE 24 – Forme du nez avec une photo de face

On transforme la classe Model de la façon suivante :

- 1 objet FaceModel
- 3 objets Individu : chacun est initialisé avec les différentes photos

5. RÉSULTATS

- 3 objets Rendu : chacun convergera vers la condition de rendu de l'image qui lui est associée

Avec cette nouvelle classe, on optimise selon la perte suivante :

$$Loss = w_l.LossLeft + w_m.LossMiddle + w_r.LossRight$$

avec $w_l = 1, w_m = 5, w_r = 1$

On donne plus d'importance au visage du milieu car dans notre façon de faire, il s'agit du visage où l'image est prise de face.

Les images 25 ci-dessous montrent les résultats obtenus après 1000 itérations, où chaque itération correspond à l'optimisation en considérant la perte globale précédente.



FIGURE 25 – Résultats avec 3 entrées

Le résultat final 26 est satisfaisant car on reconnaît la personne à partir notre reconstruction. De plus si on s'intéresse au détail, le nez, la bouche et l'arcade sont proches de la forme réelle comme en témoigne la figure 26.

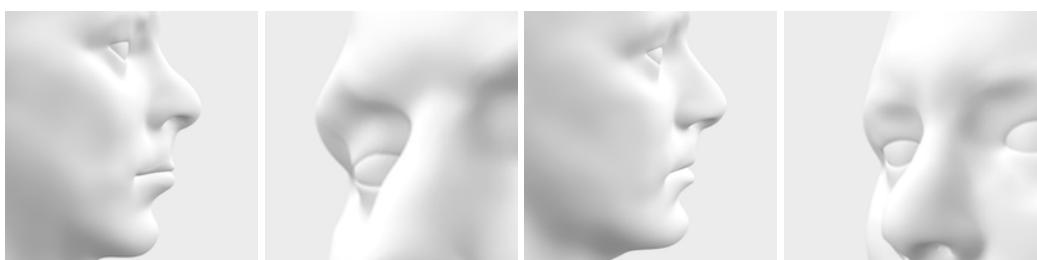


FIGURE 26 – Comparaison de la forme de l'arcade et du nez (gauche pour une entrée et droite pour trois entrées)

Dans cet exemple là, nous avons donc pu améliorer la forme de notre reconstruction. Il apparaît alors que l'utilisation de 3 caméras aide à la l'amélioration de certaines zones qui ne peuvent bien converger avec une simple photo de face.

5.2.3 • PROBLÈME CONVERGENCE SUR DES VISAGES ASIATIQUES ET AFRO-AMÉRICAIN

Les exemples des figures 28 et 27 montrent une forme globalement reconstruite. Celle d'Obama par exemple correspond à celle de la photo lorsqu'on les compare. Mais en cachant la photo de référence, on voit que cela n'est pas Barack Obama mais une version de Barack Obama "caucasien" avec une peau peu foncée. Le nez et la bouche ne sont pas identiques à la photographie. Cela est lié à notre base de données qui a été faite sur une population suisse. Ainsi, elle est trop restrictive pour pouvoir se généraliser à toutes les formes et textures de visage qui existent.

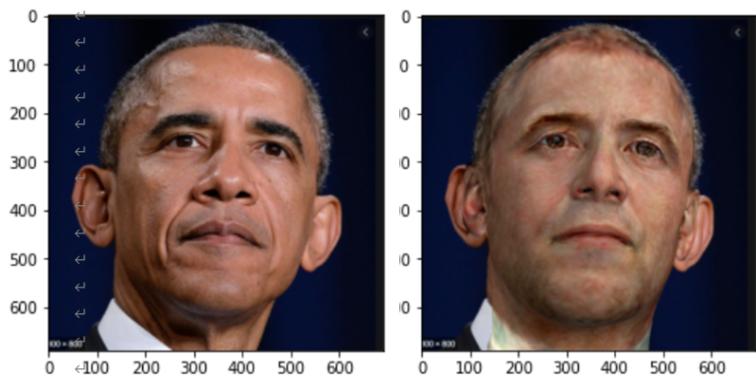


FIGURE 27 – Barack Obama : L'image initiale (à gauche) et l'image superposée (à droite)

On retrouve cela sur les visages asiatiques comme en témoigne la figure 28.

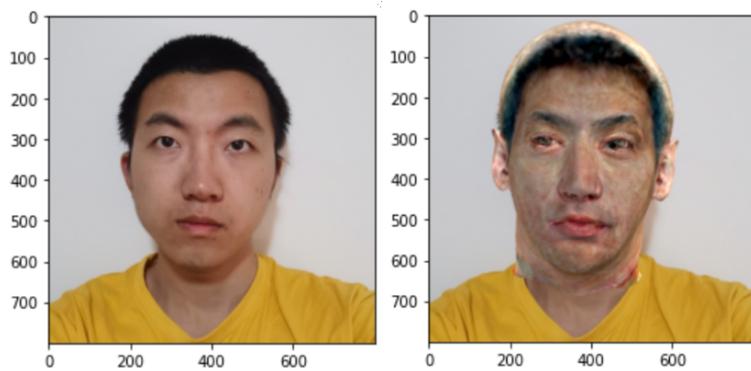


FIGURE 28 – Yutong : L'image initiale (à gauche) et l'image superposée (à droite)

On peut conclure que le visage généré est correct dans sa globalité du point de vue de la forme et de la texture. Mais c'est évidant que certains détails éthiques sont changés donnant alors une visage plutôt comme une personne blanche. Par exemple le nez est plus haut, moins plat et les os de sourcil sont plus hauts.

5.3 AMÉLIORATIONS POSSIBLES

Notre algorithme marche dans de nombreuses situations mais certains points posent problème. A travers nos essais nous avons repéré des limitations de notre algorithme qui peuvent faire l'objet d'une amélioration pour le rendre plus robuste dans son utilisation.

5.3.1 • LES CONTOURS

Le premier concerne les contours de la photo appelé "image_contour". Nous dessinons à la main avec un logiciel comme paint les contours du visage ou alors en utilisant un algorithme qui segmente automatiquement. Cet algorithme est loin d'être le plus robuste car les contours ne sont pas parfaits suivant le fond et le visage. Ainsi, on utilisait principalement des contours faits à la main.

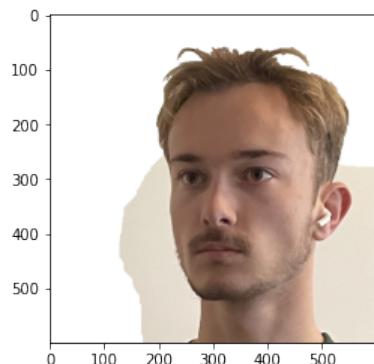


FIGURE 29 – Exemple de mauvais contour automatique

Dans notre modèle 3D, le cou est présent et souvent les contours faits coloriaient le cou en blanc. Les images de Nicolas Cage montrent que le contour blanc a été approché par notre modèle. Or avec le PCA, tout est corrélé dans notre visage donc ce changement local entraîne un changement global. C'est pourquoi le visage apparaît rougeâtre par rapport au visage de référence. Une mauvaise segmentation ne concerne pas uniquement le cou comme peut le montrer la photo ci-dessous. Et le problème est que la texture donne des informations sur la forme.

On a pensé à plusieurs façons de contrer ce problème. La première serait d'utiliser un masque à partir de l'image de rendu et non pas un masque à partir de l'image de référence. Cela éviterait de segmenter à la main ou à l'aide d'un algorithme l'image détournée. Le problème que cela pose concerne la forme finale.

5. RÉSULTATS



FIGURE 30 – Exemples du problème de contour

5.3.2 • QUALITÉ DE LA PHOTO EN ENTRÉE

Nous avons testé notre algorithme sur une photo de basse résolution pour voir la qualité du résultat que cela va produire. Nous nous doutions que plus l'image est petite et moins l'image contient d'informations sur la texture et donc la forme. La comparaison entre le résultat obtenu et l'image de référence montre que le résultat est satisfaisant mais loin d'être parfait. En particulier le nez et le contour de la bouche ne sont pas bien reconstruits. Déjà que pour des photos de qualité correcte, environ 900 par 900 pixels, ces zones ont du mal à être reconstruites alors pour des photos de petites tailles cela n'est pas possible.

Dans le cas de la reconstruction à partir de trois photos, des photos de mauvaises qualités entraînent un résultat bon dans sa globalité mais les améliorations locales observées précédemment ne sont pas toujours présentes. La forme de nez n'est pas toujours la bonne car la précision des landmarks n'est pas suffisante.

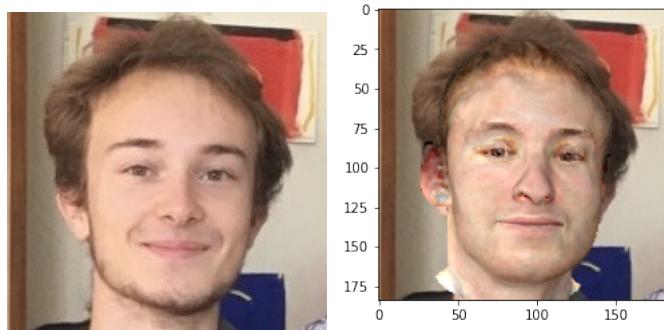


FIGURE 31 – Résultat pour une entrée de basse qualité

6

CONCLUSION

En conclusion, on a réussi le but principal de ce projet, à savoir la reconstruction de visage 3D à partir d'une image. La qualité de cette reconstruction dépend fortement de l'image initiale, du contour et de la personne sur la photo car notre base de données n'est pas assez diversifiée. Donc il y a encore des améliorations possibles de ce projet.

Avec les résultats simples, on a pu atteindre les premiers objectifs intermédiaires comme :

- l'identification des landmarks
- la mise en correspondance des points
- la reconstruction de la forme du visage

On a comparé à une approche différente qu'est Scalismo mais nous n'avons pas eu le temps d'implémenter nous même l'algorithme. On a jugé que celui-ci était une perte de temps car de toute manière il serait moins efficace en temps de convergence et en qualité de solution que notre méthode basique.

Puis nos diverses améliorations de notre algorithme ont permis de répondre à l'objectif d'« Amélioration de la forme ». Nous avons en effet étudié l'amélioration liée à l'utilisation de 3 photos au lieu d'une et l'augmentation du nombre de paramètres, en passant par l'ajout d'un nouveau paramètre et d'une optimisation différente.

Comme annoncé dans le rapport intermédiaire, l'« Amélioration de la forme grâce à la texture » a été laissé tombée car cela était trop différent de ce qui avait été fait avant. De plus, il était plus intéressant de tester en profondeur l'algorithme fait précédemment. Ainsi on peut voir cette approche comme étant le prolongement normal de notre projet. On a vu que la texture permettait d'obtenir des détails sur la forme. Cela se rapprocherait du **papier de recherche** qui infère à partir d'une texture, une texture améliorée avec des imperfections de la peau. On peut donc imaginer une approche similaire qui ajouterait les profondeurs des imperfections de la peau à notre structure 3D.

RÉFÉRENCES

- [1] Basel face model 2019. <https://faces.dmi.unibas.ch/bfm/bfm2019.html>.
- [2] Basel face model (pre-trained statistical shape model). <https://faces.dmi.unibas.ch/bfm/>.
- [3] D3dfacs (visages en 3d). <http://www.cs.bath.ac.uk/~dpc/D3DFACS/>.
- [4] dlib. http://dlib.net/face_landmark_detection.py.html.
- [5] Détection de landmarks. <http://blog.dlib.net/2014/08/real-time-face-pose-estimation>.
- [6] Face warehouse dataset (visages en 3d). <http://kunzhou.net/zjugaps/facewarehouse/>.
- [7] Génération d'images de visages à partir d'un statistical shape model et fitting de pmms. <https://github.com/Yadiraf/face3d>.
- [8] Modèle pré-entraîné d'estimation de forme de visage. <https://github.com/Yadiraf/PRNet>.
- [9] Modèle type "deep learning" de régression de visages 3d (pré-entraîné). <https://arxiv.org/abs/1905.06817>.
- [10] Opencv. <https://opencv.org/>.
- [11] Photorealistic facial texture inference using deep neural networks. <https://arxiv.org/pdf/1612.00523.pdf>.
- [12] Pytorch. <https://pytorch.org/>.
- [13] Pytorch3d. <https://pytorch3d.org/>.
- [14] Scalismo homepage. <https://scalismo.org/>.
- [15] Tutoriel sur le deep learning. <https://gravis.dmi.unibas.ch/PMM/lectures/ssm/>.
- [16] Tutoriel sur le machine learning. <https://www.coursera.org/learn/machine-learning>.
- [17] Tutoriel sur les probabilistic morphable models (statistical shape modeling et probabilistic shape fitting). <https://gravis.dmi.unibas.ch/PMM/>.
- [18] Wikipédia de scala. [https://fr.wikipedia.org/wiki/Scala_\(langage\)](https://fr.wikipedia.org/wiki/Scala_(langage)).
- [19] et al. A.S.Jackson, A.Bulat. Large pose 3d face reconstruction from a single image via direct volumetric cnn regression. In *Proceedings of 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2017.

- [20] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3d face reconstruction with weakly-supervised learning : From single image to image set. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 285–295, 2019.
- [21] Baris Gecer, Stylianos Ploumpis, Irene Kotsia, and Stefanos Zafeiriou. Ganfit : Generative adversarial network fitting for high fidelity 3d face reconstruction. pages 1155–1164, 06 2019.
- [22] et al. P. Paysan, R. Knothe. A 3d face model for pose and illumination invariant face recognition. In *Proceeding of the sixth IEEE International Conference on Advanced Video and Signal Based Surveillance.*, 2009.
- [23] et al. S. Saito, T. Simon. Pifuhd : Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, 2020.
- [24] et al. S. Sanyal, T. Bolkart. Learning to regress 3d face shape and expression from an image without 3d supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, 2019.
- [25] V. Kazemi J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, 2014.