Rapport IM06

Sacha Khosrowshahi, Nathan Rouillé, Aymane Hamdaoui

1 Introduction

Parmi les architectures les plus performantes en génération d'images réalistes styleGAN2 [Karras et al., 2020] se place comme un choix attrayant, en offrant un contrôle fin sur les attributs visuels de par les possibles manipulations du code latent qu'il produit. Le but de ce projet était pour nous de voir différentes méthodes de manipulation de ce code latent afin de trouver des manières contrôlées de modifier les attributs et les caractéristiques des images générées par styleGAN2 [Karras et al., 2020].
Nous avons exploré trois approches complémentaires et nous avons exploré une piste d'approfondissement en partant d'images réelles :

- **styleGAN2 [Karras et al., 2020]** et son mécanisme de mélange de styles (style mixing), qui permet de combiner les caractéristiques globales et locales de plusieurs visages.
- **GANSpace [Härkönen et al., 2020]**, une méthode basée sur l'analyse en composantes principales (PCA) de l'espace latent W, visant à identifier des directions de variation interprétables.
- **InterfaceGAN [Shen et al., 2020]**, qui repose sur l'entraînement de classifieurs linéaires pour isoler des directions correspondant à des attributs sémantiques spécifiques.
- **projection d'images réelles dans l'espace latent** de styleGAN2 [Karras et al., 2020]. Ce processus consiste à retrouver, pour une image donnée, le vecteur latent w qui permet de la reproduire au plus proche à l'aide du générateur. Une fois cette projection obtenue, il devient possible d'appliquer les manipulations latentes étudiées sur des visages réels, et non plus uniquement sur des visages synthétiques. Cela permet des modifications d'attributs sur des images réelles.

Notre rapport propose une exploration expérimentale de ces trois méthodes, en les comparant sur leur capacité à produire des modifications cohérentes, localisées et interprétables dans l'espace latent. Nous analyserons notamment l'effet des différentes plages de couches sur le rendu visuel, l'impact des biais de l'espace latent, ainsi que la stabilité des attributs générés.

2 styleGAN2

2.1 Paramètres de génération

Lors de la synthèse d'images avec styleGAN2 [Karras et al., 2020], des paramètres permettent de contrôler la qualité, la diversité et l'étiquette :

Troncature (ψ) : Contrôle l'équilibre entre fidélité et variété en rapprochant chaque vecteur latent w du vecteur latent moyen $\bar{w}$ (correspondant au visage moyen 2). Formellement,

$$w' = \bar{w} + \psi\,(w - \bar{w}).$$

Des valeurs $\psi < 1{,}0$ réduisent les artefacts mais limitent la diversité ; $\psi = 1{,}0$ n'applique aucune troncature.

Mode de bruit : styleGAN2 [Karras et al., 2020] injecte du bruit par couche pour produire des détails stochastiques.
- `const` : bruit fixe appris (sortie déterministe).
- `random` : bruit aléatoire à chaque image (variation des textures).
- `none` : aucun bruit (images très lisses, sans détails fins).

Étiquette (conditionnement) : Pour les modèles conditionnels, un vecteur one-hot ou d'embedding c oriente la génération vers une classe ou un attribut particulier.
Dans notre cas, nous utilisons un modèle non conditionnel (FFHQ), donc cet argument vaut `None`.

(a) Bruit constant (b) Bruit aléatoire (c) Pas de bruit

FIGURE 1 – Impact du mode de bruit sur la génération. Le mode constant fournit les meilleurs détails, le mode aléatoire offre une variation intéressante, et l'absence de bruit produit des images trop lisses sans détail dénaturant l'image lorsque l'on s'approche.

FIGURE 2 – Visage moyen

(a) $\psi = 0{,}5$ (b) $\psi = 0{,}7$ (c) $\psi = 1{,}0$ (d) $\psi = 1{,}3$ (e) $\psi = 1{,}5$

FIGURE 3 – Effet de la troncature ψ sur les images générées. La diversité augmente avec ψ, au prix d'un risque accru d'artefacts.

2.2 Mélange de styles dans styleGAN2 [Karras et al., 2020]

Le mélange de styles est une technique introduite dans styleGAN2 [Karras et al., 2020] pour combiner les attributs globaux (pose, forme) d'un échantillon latent A avec les détails fins (texture, couleur) d'un échantillon B. Étant donnés deux codes latents w_A et w_B dans l'espace intermédiaire W^+, on choisit un ensemble de couches k. Pour ces couches, on injecte w_A, et pour les autres, on injecte w_B. Par exemple, pour $k = \{0, 1, \ldots, 6\}$, l'image résultante hérite de la structure globale d'A et des détails de B.

FIGURE 4 – Mélange de styles avec lignes = [1203,3405,34,902,910] et colonnes = [8023,122,67,7161], styles 0–6

FIGURE 5 – Mélange de styles avec lignes = [1203,3405,34,902,910] et colonnes = [8023,122,67,7161], styles 6–17

Détails d'implémentation. Dans nos expériences, nous avons généré deux ensembles de codes latents avec des seeds fixes, puis pour un ensemble de couches k :

$$w_{\text{mix}} = w_A, \quad w_{\text{mix}}[k] = w_B[k],$$

et synthétisé l'image via `G.synthesis(w_mix)`. Comme expliqué précédements, les premières couches transmettent la structure globale d'A (posture, expression...), tandis que les dernières transmettent les détails de B (couleurs des cheveux et habits, arrière plan...).

3 GANSpace

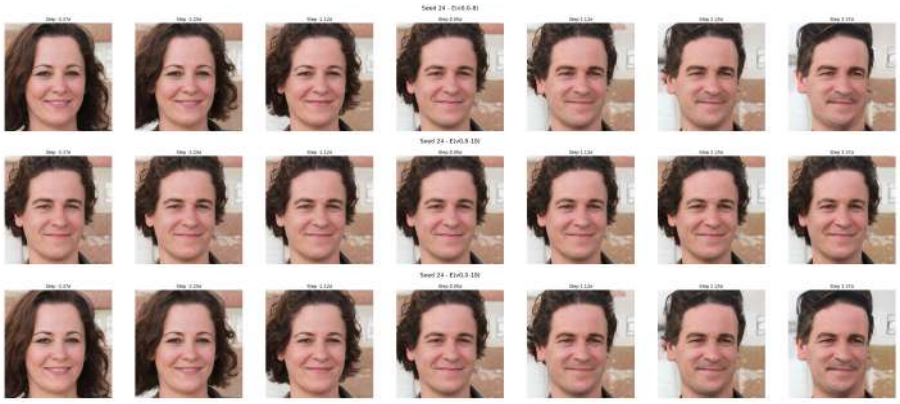
Une autre méthode proposée qui s'inscrit dans le projet de manipulation de code latent est le GANSpace [Härkönen et al., 2020]. Cette méthode propose de

faire une Analyse de composante principales sur le code latent afin d'obtenir les vecteurs qui expliquent le plus de variance dans l'espace W, puis de modifier l'étape du vecteur de style au sein de chaque couche.
En effet, pour rappel, l'architecture styleGAN2 [Karras et al., 2020] produit un vecteur latent de style w (ou w_i dans le cas de style mixing où chaque couche reçoit un vecteur différent) et le fournit à chaque couche du générateur.
Le vecteur de style est alors modifié pour un nombre de couche séléctionné comme suit :

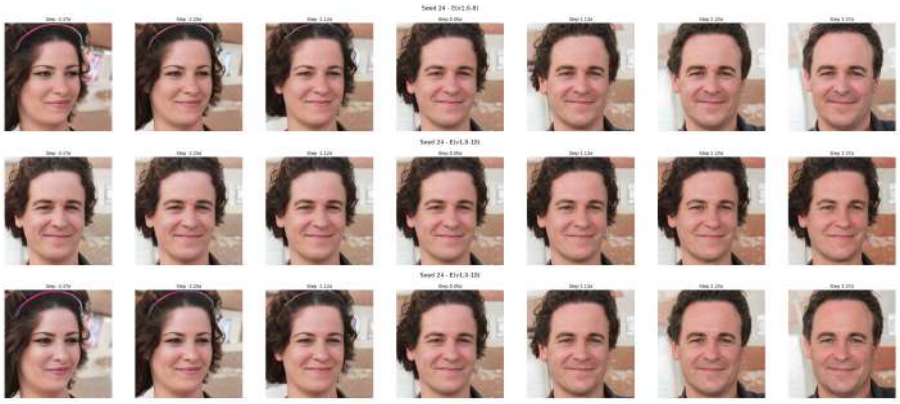
$$\mathbf{w}' = \mathbf{w} + V\mathbf{x} \tag{1}$$

- $\mathbf{w}$: vecteur latent original dans l'espace $\mathcal{W}$ de styleGAN2 [Karras et al., 2020] (sortie du réseau de mapping).
- V : matrice des composantes principales (issues de l'analyse PCA de $\mathcal{W}$).
- $\mathbf{x}$: vecteur de coefficients scalaires définissant l'intensité de chaque direction.
- $\mathbf{w}'$: vecteur latent modifié résultant de la manipulation.

Ainsi en faisant cette manipulation nous pouvons directement voir l'influence qu'a chaque couche sur le résultat final et sur la sémantique des visages. Au delà de ça le fait d'ajouter au vecteur du style un des vecteurs principaux de la PCA va nous permettre de voir quel vecteur influe sur quel attribut et à quelle intensité.
Après implémentation nous itérons sur les différentes parties du réseau générateur et sur les différents vecteurs de la PCA afin de voir l'évolution avec un pas sur les différentes directions de la PCA.

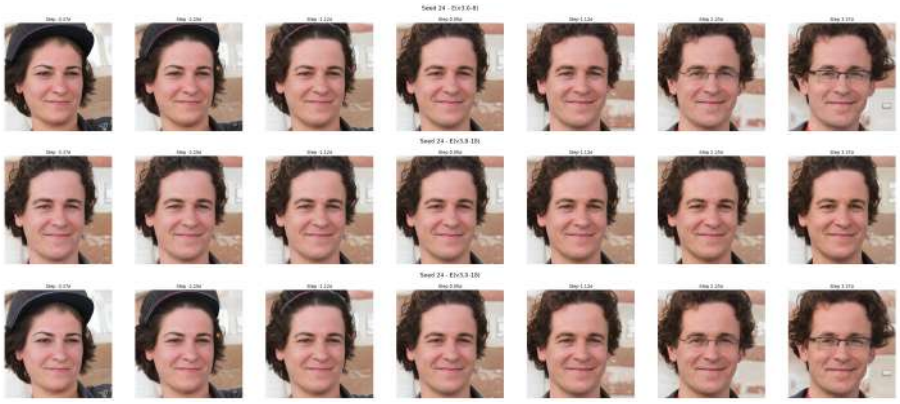


(a) Évolution selon la direction v_0



(a) Évolution selon la direction v_1

(b) Évolution selon la direction v_2



(c) Évolution selon la direction v_3

FIGURE 7 – Manipulations latentes sur la seed 24 selon différentes directions v_k extraites par PCA (GANSpace). Chaque ligne correspond à une plage de couches modifiée.

3.1 Intérpretation des résultats

Ainsi nous avons plot ici des déplacements pour chaque direction choisie de la PCA et pour chaques couches du générateur choisies. L'intérprétation, comme le décrit l'article GANSpace [Härkönen et al., 2020] est empirique. En effet ce n'est qu'une fois les tests faits que nous pouvons determiner quelle direction influe sur quelle caractéristique ou trait facial.

Effets des différentes plages de couches. L'édition est appliquée à trois intervalles de couches distincts : **[0–8]**, **[8–18]** et **[0–18]**, pour un même vecteur de direction et le même seed.

- **Layers 0–8 :** cette plage de couches correspond aux premières couches du réseau de synthèse et affecte principalement la structure globale et les traits sémantiques majeurs du visage. On observe un changement marqué de l'identité : modification de la forme du visage, du regard, de l'expression faciale, ainsi que l'apparition ou disparition de la barbe. Le fond reste inchangé, ce qui confirme le rôle de ces couches dans la modélisation de la géométrie du visage sans influencer le style ou l'arrière-plan.
- **Layers 8–18 :** ces couches interviennent à des niveaux plus profonds du réseau, et modifient surtout le style visuel. L'identité reste globalement constante, mais des changements apparaissent au niveau du fond, de l'éclairage et de la texture (on aperçoit une couleur plus vive pour le fond par exemple). On note également l'apparition d'artefacts visuels dans les extrêmes de la direction ($\pm 3\sigma$).
- **Layers 0–18 :** l'édition sur toutes les couches combine les effets précédents : transformation complète du visage (identité, expression, pilosité) avec des variations de fond et de texture. Toutefois, cette approche engendre également davantage d'instabilité visuelle, avec l'apparition de distorsions comme on le voit sur le visage deformé aux valeurs ($\pm 3\sigma$).

Interprétation selon SpaceGAN. Ces observations appuyent les résultats du papier *SpaceGAN*, qui montre que l'espace latent de styleGAN2 [Karras et al., 2020] est organisé hiérarchiquement : les couches basses contrôlent la sémantique (âge, genre, forme du visage), tandis que les couches hautes modulent le style, les couleurs et le contexte. L'édition par plage de couches permet ainsi une manipulation plus fine et plus contrôlée que l'édition globale.

Conclusion. On peut grâce à ces différents tests établir le lien direct entre certaines directions et des attributs, cependant contrairement à InterfaceGAN [Shen et al., 2020] nous ne pouvons pas établir des liens uniques : v_0 -> genre ; v_1 -> genre et orientation de la tête ; v_2 -> orientation de la tête sens inverse et genre ; v_3 -> accessoires et genre.
Cette méthode est donc un bon moyen de manipuler l'architecture du générateur et du code latent pour faire de la modification, cependant elle ne paraît pas aussi controlée et précise que l'on veuille.

4 InterfaceGAN

InterfaceGAN [Shen et al., 2020], introduit par Shen *et al.*, est une méthode qui exploite l'espace latent d'un GAN pour offrir un contrôle interactif des attributs sémantiques. L'idée centrale consiste à entraîner, pour chaque attribut d'intérêt (âge, genre, sourire, etc.), un classifieur linéaire dans l'espace intermédiaire W. Le vecteur normal de ce classifieur définit une direction de modification : en déplaçant un code latent le long de cette direction, on augmente ou diminue l'attribut correspondant dans l'image générée ou projetée. InterfaceGAN [Shen et al., 2020] fournit ainsi une interface simple et intuitive pour explorer et manipuler visuellement les variations sémantiques au sein des images produites par un GAN. En particulier nous utiliserons l'implémentation fournie de styleGAN2 [Karras et al., 2020] pour la suite.

4.1 Détails d'implémentation

Dans notre pipeline, nous commençons par échantillonner 10 000 vecteurs $z \in \mathbb{R}^{512}$ distribués selon $\mathcal{N}(0, I)$, que nous passons dans le *mapping network* de styleGAN2 [Karras et al., 2020] pour obtenir leurs codes intermédiaires $W_s \in \mathbb{R}^{10\,000 \times L \times 512}$ (à noter que nous n'utilisons pas de vecteur de style particulier donc toutes les couches sont les même et afin d'alléger la taille de nos données nous ne gardons que la première couche des vecteurs w ce qui résults alors en un tableau $Ws \in \mathbb{R}^{10\,000 \times 512}$). Chaque code $w_i \in \mathbb{R}^{L \times 512}$ (on récupère un vecteur w' depuis W_s qui est répétée L fois) est ensuite synthétisé en une image via `G.synthesis`, produisant 10 000 visages correspondants. Nous utilisons un classifieur pré-entraîné sur les attributs suivants (issus de la base CelebA) :

```
0: 5_o_Clock_Shadow, 1: Arched_Eyebrows, 2: Attractive, 3: Bags_Under_Eyes,
4: Bald, 5: Bangs, 6: Big_Lips, 7: Big_Nose, 8: Black_Hair, 9: Blond_Hair,
10: Blurry, 11: Brown_Hair, 12: Bushy_Eyebrows, 13: Chubby, 14: Double_Chin,
15: Eyeglasses, 16: Goatee, 17: Gray_Hair, 18: Heavy_Makeup, 19: High_Cheekbones,
20: Male, 21: Mouth_Slightly_Open, 22: Mustache, 23: Narrow_Eyes, 24: No_Beard,
25: Oval_Face, 26: Pale_Skin, 27: Pointy_Nose, 28: Receding_Hairline,
29: Rosy_Cheeks, 30: Sideburns, 31: Smiling, 32: Straight_Hair, 33: Wavy_Hair,
34: Wearing_Earrings, 35: Wearing_Hat, 36: Wearing_Lipstick,
37: Wearing_Necklace, 38: Wearing_Necktie, 39: Young
```

Pour chaque attribut sélectionné, nous récupérons les scores de classification sur les 10 000 images. Nous trions ces scores et retenons les 2% de plus hautes valeurs comme échantillons positifs et les 2% de plus basses comme négatifs. Sur cet ensemble réduit (4% des données), nous entraînons un SVM linéaire pour séparer positifs et négatifs. Le vecteur normal du SVM, définit alors la *direction* à appliquer dans l'espace W pour augmenter ou diminuer l'attribut considéré. A noter que nous allons réduire l'espace des attributs de façon arbitraire aux attributs qui nous semblait être les plus interprétables visuellement.

4.2 Importance de l'orthogonalisation

Une fois que ces directions sont obtenues il est simple de générer de premiers résultats. Nous remarquerons alors le fort biais qui peut apparaître. En particulier, certains attributs comme "Wearing Lipstick" sont fortement biaisé. On remarque sur la Figure8 que sans orthonormaliser nos directions avec un procédé comme Gram-Shmidt, en se déplaçant sur la direction correspondante, nous affectons d'autres attributs tels que le genre. Il est donc nécessaire de chercher à décorréler nos directions. La Figure9 illustre de manière quantitative cet impacte. Malgré cela, il reste un biais intrinsèque au modèle qui est difficile à supprimer.

(a) Sans orthonormalisation

(b) Avec orthonomalisation

FIGURE 8 – Comparaison de l'impact de l'orthonormalisation par Gram–Schmidt : en haut, l'attribut Rouge à lèvre appliqué tel quel ; en bas, après avoir rendu les directions orthogonales.

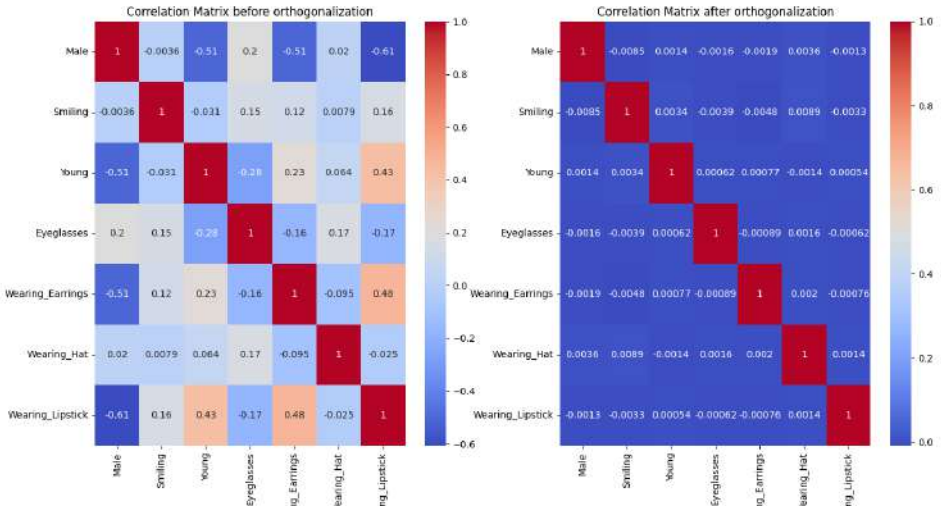


FIGURE 9 – Matrices de corrélations sur un sous-ensemble d'attributs

4.3 Espace Z ou espace W ?

Le papier d'InterfaceGAN [Shen et al., 2020] explique qu'il est possible de faire le même travail mais dans l'espace Z au lieux de W. Malgré tout, il est mentionné que Z est "moins séparable" et que les résultats sont donc meilleurs dans l'espace W. Dans un premier temps nous pensions que cela signifiait que les vecteurs normaux de Z étaient moins faciles à décorréler. Nous avons donc reproduit notre travail précédent afin d'obtenir la Figure10. Cette dernière nous a alors étonnée car il nous semblait que les résultats étaient semblables voir meilleurs.

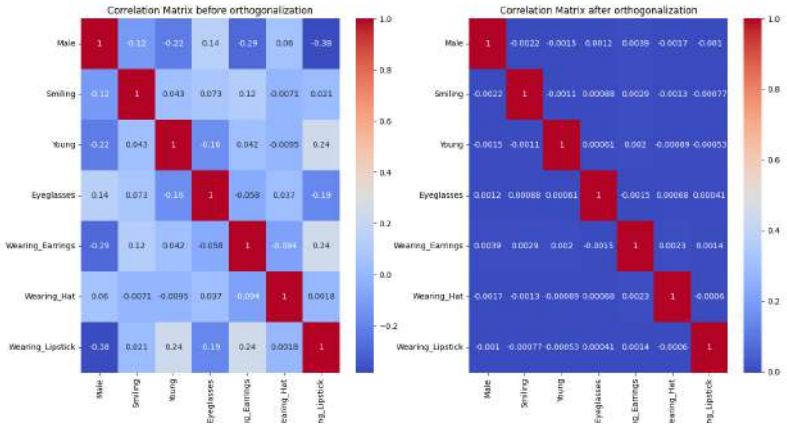


FIGURE 10 – Matrices de corrélations dans Z sur un sous-ensemble d'attributs

Nous avons alors compris que le problème se trouvait ailleurs. Le papier mentionnait la séparabilité vis à vis des attributs sémantiques. Nous avons alors regardé les résultats de nos classifiers respectivement dans Z et dans W. Ces résultats sont illustrés dans la Figure11 et tendent vers la même conclusion que celle énoncée par le papier.

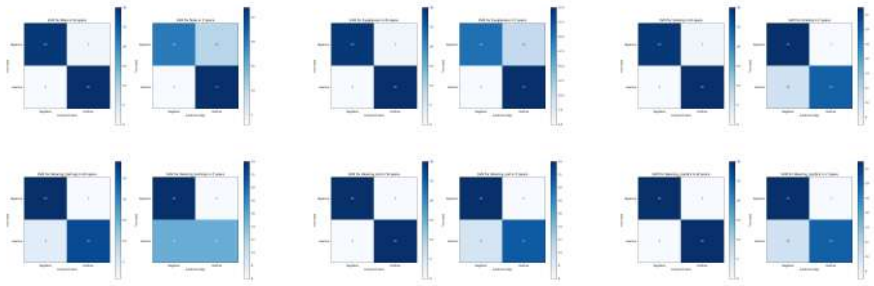


FIGURE 11 – Comparaison des classifiers selon les espaces avec à gauche dans l'espace W et à droite dans l'espace Z.

4.4 Résultats

Les résultats obtenus sont pour la majorité très convaincant. Malgré tout, certains biais apparaissent clairement. Certains de ces biais sont prévisibles à l'aide de la matrices de corrélations. Par exemples il était possible de prévoir que le port d'un chapeau était corrélé à la présence de lunettes (14). D'autres sont inattendus (15) car nous n'avons pas pu utiliser d'attributs sémantiques afin de supprimer le biais. A noter que ces biais sont dus au jeux d'entrainement. En particulier nous avons utiliser un générateur entrainé sur ffhq qui est un

jeux de donnée mal balancé d'après [Maluleke et al., 2022] qui met en lumière particulièrement les proportions non équilibrés entre personnes blanches et non blanches. Cela se voit en particulier sur le visage moyen qui est blanc avec les yeux verts. Ces déséquilibres influent donc directement nos générations en créant des biais difficile à supprimer.

FIGURE 12 – Attribut Genre

FIGURE 13 – Attribut Lunettes

FIGURE 14 – Attribut Casquette

FIGURE 15 – Attribut Boucle d'oreille

5 Projection d'images réelles

5.1 Méthode d'inversion de styleGAN2 [Karras et al., 2020]

Pour appliquer nos manipulations à des images réelles, nous devons retrouver, pour chaque image $x \in \mathbb{R}^{H \times W \times 3}$, un code latent $\hat{w} \in \mathcal{W}$ tel que $G(\hat{w}) \simeq x$. Nous avons utilisé l'implémentation projector.py qui suit la procédure d'optimisation proposée dans l'Appendix D de styleGAN2 [Karras et al., 2020] :

1. Initialisation par la moyenne $\bar{w}$ des points générés dans l'espace W.
2. Optimisation (Adam) sur 1000 itérations de la perte

$$\mathcal{L}(\hat{w}) = \underbrace{\text{LPIPS}\big(x, G(\hat{w})\big)}_{\text{distance perceptuelle}} + \underbrace{\alpha \sum_{i,j} L_{i,j}}_{\text{régularisation bruit}}$$

où la distance LPIPS (Learned Perceptual Image Patch Similarity) mesure la similarité perceptuelle entre les 2 images en comparant leurs activations dans un CNN
et les $L_{i,j}$ sont des termes de régularisation du bruit à différentes échelles dans la pyramide de résolution du bruit $n_{i,j}$

Avec les paramètres par défaut (1000 itérations) sur le GPU3 l'inversion complète d'une image prend $\approx 2\,\text{min}$.

5.2 Qualité de projection

La figure 16 juxtapose plusieurs couples *(image réelle, reconstruction après projection)*. Nous constatons qu'il y a parfois des échecs de reconstruction : des images visuellement très différentes et une distance LPIPS plus élevée. Ces erreurs peuvent s'expliquer par de distribution différentes de celles des images FFHQ (ici orientation et cadrage différents).

LPIPS 0.35 0.36 0.39 0.44

FIGURE 16 – Projections réelles : deux bons exemples (gauche) et deux échecs (droite). Ligne 1 : photo originale ; ligne 2 : reconstruction $G(\hat{w})$.

5.3 Amélioration InterfaceGAN : sélection automatique du pas α

Comme nous l'avons vu dans InterfaceGAN 4, une fois les directions d'attributs $\{n_j\}_j$ rendues orthogonales 4.2, la modification d'un visage revient à déplacer son code latent $w \in \mathcal{W}$ selon $w' = w + \alpha\, n_j$. La difficulté est de trouver *pour chaque image* et chaque attribut un pas α :

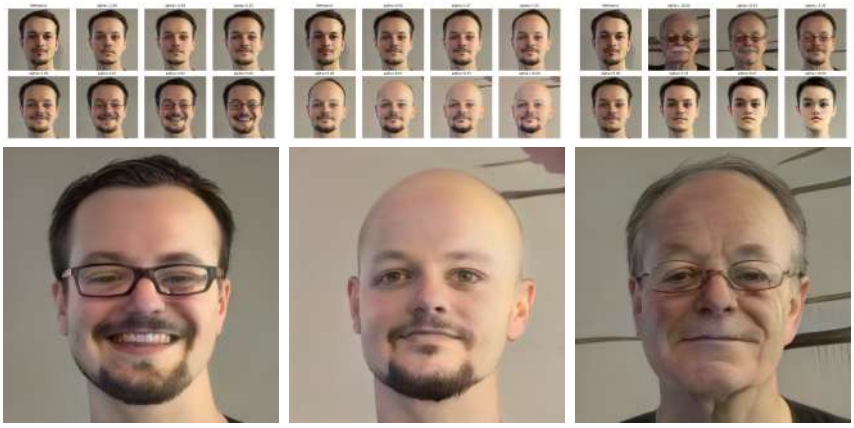
— α trop petit : l'attribut reste peu marqué ;
— α trop grand : on sort de la zone des visages plausibles (apparition d'artefacts).

Idée. Lors de l'entraînement des SVMs, nous possédons déjà les 2 % de latents dont le classifieur d'attribut est le plus élevé. Nous en tirons simplement leur centroïde $\mu_j = \frac{1}{K}\sum_{k=1}^{K} w_k^{(j)}$ et l'enregistrons. Ainsi, pour un nouveau code w nous choisissons le pas

$$\boxed{\alpha \;=\; n_j^{\top}(\mu_j - w)} \tag{1}$$

où n_j est *unitaire*. Il s'agit de la projection orthogonale de μ_j sur la droite $\{\, w + t\, n_j \mid t \in \mathbb{R}\}$. Autrement dit, nous déplaçons w du vecteur dans la direction de l'attibut qui l'amène statistiquement au plus proche des points qui partagent cet attribut.

Cette règle ne nécessite pas de boucles d'essais : l'édition est donc *complètement automatisable*, même pour la génération aléatoire d'images possédant un attribut imposé tout en restant dans la région la plus vraisemblable de $\mathcal{W}$.



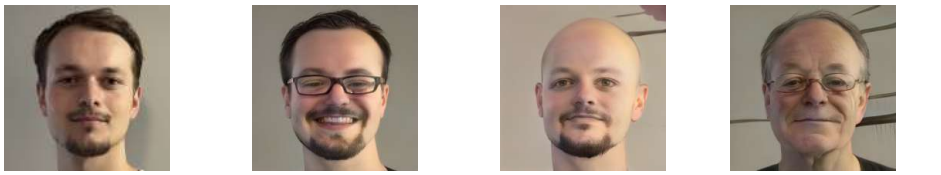
Valeur α 5.3 | 8.5 | -6.6

FIGURE 17 – Résultat de la méthode de sélection automatique de α pour plusieurs attributs (Sourire, Chauve, Âge) : résultats pour un range de valeurs de α (ligne 1) et résultat obtenu avec le α sélectionné automatiquement (ligne 2).

Comme nous pouvons voir sur la Figure 17 cette méthode a permis pour chacun des 3 attributs : Sourire, Chauve et Age de déterminer une valeur de α très plausible sans itérer sur un large range de valeurs.

5.4 Édition sémantique d'une image projetée

Une fois le code latent $\hat{w}$ de l'image projetée obtenu, nous pouvons donc appliquer la méthode d'édition InterfaceGAN [Shen et al., 2020] automatique précédente 5.3 :



Originale | Sourire (correlé à lunettes) | Chauve | Agé (correlé à lunettes)

FIGURE 18 – Édition sémantique de l'image originale

Nous observons que les méthodes précédentes appliquées à des codes latents mappés depuis un bruit comme InterfaceGAN [Shen et al., 2020] sont également applicable à des codes latents obtenus à partir d'images réelles projetées. Notamment les direction orthogonales au SVM, qui a été entraîné uniquement sur des codes générés, semblent toujours sémantiquement valable et la méthode de sélection automatique du pas grâce aux moyennes sémantiques de codes générés

fonctionne également.
Cependant, on remarque comme précédemment que la distribution des images réelles est différente de celle des images générées dans cet espace latent, ce qui entraine des comportements légèrement différents notamment sur l'enchevêtrement entre les attributs, on peut voir ici que l'attribut lunettes semble bien plus enchevêtré avec d'autres attributs qu'il ne l'était lors de la manipulation d'images générées.

Une autre propriété intéressante est la suivante : il semble que pour obtenir des résultats plus réalistes, il serait préférable de projeter l'image la plus neutre possible (cadrage, attributs, orientation du visage) afin d'être le plus proche possible de la distribution des images générées. On remarque notamment sur la Figure 19 que le résultat obtenu par édition du visage neutre en ajoutant les attributs chapeau et lunettes est plus réaliste que celui obtenu en projetant une image avec un chapeau et des lunettes. En effet bien que la composition de l'image projetée soit plus fidèle, les textures de la peau et la moustache ne sont pas du tout réaliste contrairement à l'image neutre éditée qui bien que le chapeau ait un artefact reste très réaliste.

FIGURE 19 – Comparaison du résultat obtenu par édition d'une image neutre et projection d'une image avec attributs

5.5 Détection de visages générés

Conformément à la Figure20, image issue de la publication styleGAN2 [Karras et al., 2020], qui illustre les différences de distance LPIPS entre une image et sa projection pour les images réelles et synthétiques, nous pouvons distinguer les images synthétiques et réelles par un seuillage sur $\text{LPIPS}\big(x, G(\hat{w})\big)$. Dans la publication originale le seuil se trouve aux alentours de 0.2, cependant nous avons identifié de meilleurs résultats pour un seuil de 0.3. La complexité algorithmique de la projection nécessaire à cette méthode rend notre détermination statistique d'un seuil optimal peu fiable étant donné la faible taille de notre

échantillon. L'algorithme complet :

1. inversion de l'image (Section 5.1) ;
2. calcul de LPIPS entre l'image originale et sa reconstruction après projection ;
3. décision `image synthétique` si LPIPS ≤ 0.3 et sinon réelle.

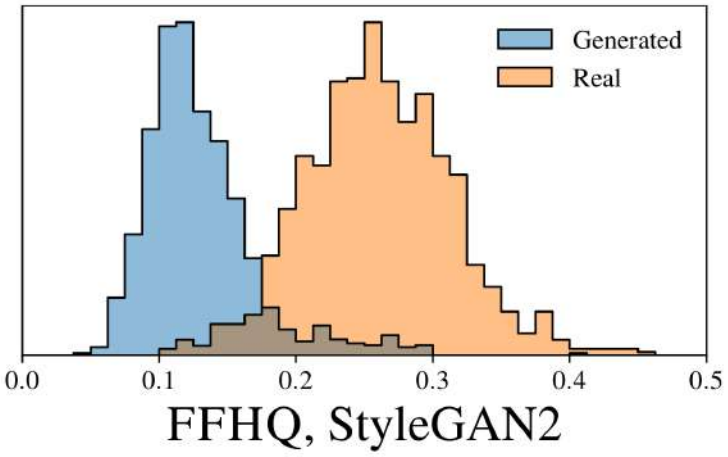


FIGURE 20 – Histogramme des distances LPIPS entre images originales et projetées pour des images synthétiques (bleu) et réelles (orange)

Nous testons maintenant cette méthode sur des images synthétiques (Figure21) puis réelles (Figure22) :

LPIPS 0.15 0.1 0.14 0.21

FIGURE 21 – Images synthétiques (ligne 1) et leurs projections (ligne 2) ainsi que leur distance LPIPS

FIGURE 22 – Images réelles (ligne 1) et leurs projections (ligne 2) ainsi que leur distance LPIPS

On remarque donc effectivement qu'avec un seuil de 0.3 on aurait ici détecté toutes les images synthétiques de visages générés.

6 Conclusion

Dans ce rapport, nous avons exploré différentes méthodes de manipulation de l'espace latent de styleGAN : le mélange de styles, GANSpace [Härkönen et al., 2020], InterfaceGAN [Shen et al., 2020], et enfin la projection d'images réelles suivie d'édition sémantique. Ces approches nous ont permis d'analyser la structure hiérarchique de l'espace W, de manipuler des attributs visuels de façon contrôlée, et d'étudier les biais et limites intrinsèques du modèle.

Chaque méthode présente des avantages spécifiques : GANSpace [Härkönen et al., 2020] permet des modifications par couches du générateur afin de visualiser des variations locales et globales, tandis qu'InterfaceGAN [Shen et al., 2020] permet une modification précise et ciblée d'attributs sémantiques grâce à des classifieurs supervisés. L'extension de ces techniques aux images réelles, via la projection, ouvre des perspectives concrètes en édition faciale et détection d'images synthétiques.

Cependant, ces manipulations révèlent également les biais du dataset d'entraînement (FFHQ), les corrélations involontaires entre attributs, et les difficultés à projeter fidèlement des images hors-distribution. Ce travail met ainsi en lumière les potentialités des GANs pour la génération et l'édition d'images, tout en soulignant les défis éthiques et techniques liés à leur usage.

Références

[Härkönen et al., 2020] Härkönen, E., Hertzmann, A., Lehtinen, J., and Paris, S. (2020). Ganspace : Discovering interpretable gan controls. In *Advances in*

Neural Information Processing Systems (NeurIPS), volume 33, pages 9841–9850.

[Karras et al., 2019] Karras, T., Laine, S., and Aila, T. (2019). A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410.

[Karras et al., 2020] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119.

[Maluleke et al., 2022] Maluleke, V. H., Thakkar, N., Brooks, T., Weber, E., Darrell, T., Efros, A. A., Kanazawa, A., and Guillory, D. (2022). Studying bias in gans through the lens of race.

[Pidhorskyi et al., 2023] Pidhorskyi, S., Draelos, R., Kruse, T., and Odena, A. (2023). Spacegan : Layer-wise latent space editing in stylegan2. *arXiv preprint arXiv :2303.08325*.

[Shen et al., 2020] Shen, Y., Gu, J., Tang, X., and Zhou, B. (2020). Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9243–9252.