



La Grande Ecole de l'IA & de la Data
Paris Île-de-France · Nice Côte d'Azur

Rapport GENAI

Génération de visages photo-réalistes avec contrôle d'attributs

MSc AI for Business

Auteurs :

Omar EL ALAMI · Ilyes SAIS · Franck BONI · Abderazak BETTAYEB

Ahmed GUEROUANI · Othmane BAMBA · Siham KIARED · Aimé Magloire NDJENG

Octobre 2025

Ce document présente une revue critique des méthodes de génération de visages photo-réalistes avec contrôle d'attributs, en mettant l'accent sur les approches GAN et diffusion, ainsi que sur les considérations éthiques et techniques.

Mots-clés : StyleGAN, GAN, Diffusion, Contrôle d'attributs, Génération d'images, Intelligence Artificielle, Éthique

Table des matières

Note de synthèse	3
1 Introduction	3
2 Fondements théoriques	3
2.1 Réseaux antagonistes génératifs (GAN)	3
2.2 Modèles de diffusion	4
2.3 Conditionnement et guidage	4
3 Méthodes de génération de visages	4
3.1 Approches basées sur les GAN	4
3.2 Approches basées sur la diffusion	4
4 Techniques de contrôle d'attributs	5
4.1 Conditionnement par labels	5
4.2 Disentanglement et directions latentes	5
4.3 Inversion et édition pilotée par texte	5
4.4 Guidage externe	5
5 Jeux de données et préparation	5
5.1 Principaux jeux de données	5
5.2 Préparation du dataset pour le projet	6
6 Métriques d'évaluation	6
7 Implémentation et expérimentations StyleGAN3	6
7.1 StyleGAN2 conditionnel : architecture et pipeline	6
7.2 Hyperparamètres d'entraînement	7
7.3 StyleGAN3 alias-free et comparaison qualitative	7
7.4 Expérimentations StyleGAN3 sur FairFace	7
7.4.1 Architecture spécifique et paramètres	8
7.4.2 Résultats quantitatifs	8
7.4.3 Analyse qualitative	10
8 Implémentation et expérimentations StyleGAN2 (V1/V2)	12
8.1 Description globale du pipeline	12
8.2 Hyperparamètres d'entraînement (StyleGAN2-V1/V2)	14
8.2.1 Justification des choix (V1/V2)	14
8.2.2 Temps et ressources	15
8.3 Réécriture « StyleGAN2 conditionnel : architecture et pipeline »	15
8.4 Courbes et comparaison V1 vs V2	16
8.5 Lecture des courbes (synthèse)	16
8.6 Résultats quantitatifs	17
8.7 Analyse des courbes	17
8.8 Attributs conditionnels	17
9 Déploiement et mise à disposition du générateur StyleGAN3	17

10 Implémentation, déploiement et résultats StyleGAN2 (v1 vs v2)	21
10.1 Description globale et périmètre	21
10.2 Architecture de code et checkpoints	21
10.3 Démo locale Gradio	22
10.4 Résultats qualitatifs (évaluation humaine)	22
10.5 Comparaison quantitative synthétique	23
10.6 Courbes d'entraînement : v1 vs v2	23
10.7 Analyse des courbes	24
10.8 Remarques finales	24
11 Analyse critique comparative des paradigmes de génération StyleGAN3	25
11.1 Qualité et diversité	25
11.2 Contrôle des attributs et désentrelacement	25
11.3 Stabilité et robustesse d'entraînement	25
11.4 Latence, déploiement et coût opérationnel	25
11.5 Biais et équité	26
11.6 Impact énergétique et écologique	26
11.7 Limites connues et pistes d'amélioration	26
12 Analyse critique comparative des paradigmes de génération StyleGAN2 (v1 vs v2)	26
12.1 Qualité et diversité	27
12.2 Contrôle des attributs et désentrelacement	27
12.3 Stabilité et robustesse d'entraînement	27
12.4 Latence, déploiement et coût opérationnel	27
12.5 Biais et équité	27
12.6 Impact énergétique et écologique	27
12.7 Limites connues et pistes d'amélioration	28
Références	29
Annexes	30
A Mini-approche StyleGAN2 conditionnel (extrait de code)	30

Note de synthèse

La synthèse de visages photo-réalistes a franchi un cap majeur au cours de la dernière décennie, portée par deux familles de modèles : les réseaux antagonistes génératifs (GAN) et les modèles de diffusion. Au-delà du réalisme, un enjeu central est le **contrôle des attributs** (âge, genre, expression, accessoires, etc.), condition indispensable à de nombreuses applications (avatarisation, retouche, essais cosmétiques, création visuelle). Ce document propose une revue critique des fondements théoriques, des approches de pointe, des jeux de données, des métriques d'évaluation et des défis éthiques, en mettant l'accent sur les méthodes qui permettent d'**imposer des contraintes sémantiques** tout en préservant la fidélité visuelle.

1 Introduction

La génération d'images par intelligence artificielle, et en particulier de visages synthétiques réalistes, est devenue l'un des domaines les plus dynamiques de l'apprentissage profond depuis l'introduction des **GAN** par Goodfellow et *al.* [1]. Les progrès récents des **modèles de diffusion** ont encore élevé le niveau de qualité, avec des images haute résolution rivalisant avec l'état de l'art [18].

La problématique abordée est celle du **contrôle d'attributs** : comment spécifier de manière fiable des caractéristiques cibles (par exemple « femme âgée, cheveux blonds, souriante, lunettes ») tout en conservant **réalisme, cohérence identitaire** et **diversité**? Pour fixer les idées, on peut se représenter un **studio photo virtuel** : l'utilisateur ajuste des « curseurs sémantiques » (expression, âge, coiffure), et le modèle génératif se comporte comme un photographe numérique obéissant à ces directives.

La suite est organisée comme suit : la Section 2 présente les fondements théoriques des GAN et des modèles de diffusion. La Section 3 décrit les principales méthodes de génération de visages. La Section 4 discute des techniques de contrôle d'attributs. La Section 5 s'intéresse aux jeux de données et à leur préparation. La Section 6 détaille les métriques d'évaluation. La Section 7 présente l'implémentation concrète et les expérimentations du projet. Les Sections 11 à ?? proposent une analyse critique, examinent les aspects éthiques et concluent sur les perspectives de recherche.

2 Fondements théoriques

2.1 Réseaux antagonistes génératifs (GAN)

Le principe adversarial met en jeu un **générateur** et un **discriminateur** dans un jeu à somme nulle : le premier tente de produire des échantillons indiscernables du réel, le second d'en déceler la nature artificielle [1]. Malgré leurs succès, les GAN ont longtemps souffert d'**instabilités d'entraînement** (oscillations, *mode collapse*), ce qui a motivé de nombreuses variantes améliorant la convergence et la résolution. Des jalons importants incluent **DCGAN**, la **croissance progressive** de la résolution (Progressive GAN) et la série **StyleGAN**, qui a établi de nouveaux standards de qualité sur les visages à très haute résolution.

2.2 Modèles de diffusion

Les modèles de diffusion (DDPM, *score-based*) apprennent à **inverser** un processus de bruitage progressif pour remonter du bruit vers l'image [18]. La génération s'effectue en **plusieurs étapes** de débruitage, donnant des images d'une qualité remarquable et une excellente **couverture de la distribution** (faible risque de *mode collapse*). Leur principal coût est **informatique** : de nombreuses itérations par image et des modèles lourds, surtout en haute résolution. L'architecture **Latent Diffusion** (LDM) déplace la diffusion dans l'**espace latent** d'un autoencodeur, réduisant drastiquement le coût tout en préservant les détails.

2.3 Conditionnement et guidage

Le **conditionnement explicite** via des étiquettes (cGAN) incorpore des variables d'attributs dans le générateur et le discriminateur [2]. Du côté diffusion, des techniques de **guidage** orientent l'échantillonnage vers les attributs cibles : *classifier guidance* (utilisation d'un classifieur externe) et *classifier-free guidance* (combinaison conditionnelle/non conditionnelle). Dans les architectures de type StyleGAN et LDM, le conditionnement peut aussi passer par la **modulation de couches** (AdaIN/FiLM) ou la **cross-attention** pour le guidage par texte.

3 Méthodes de génération de visages

3.1 Approches basées sur les GAN

La famille **StyleGAN** a introduit un **espace latent structuré** et des injections de style qui dissocient facteurs de haut niveau (identité, pose) et détails stochastiques (grain de peau, cheveux) [5, 17]. StyleGAN2 corrige les artéfacts caractéristiques de la première version et remplace la normalisation par une *démodulation des poids*. Ces modifications abaissent le **FID** sur FFHQ et produisent des visages quasi indiscernables.

L'itération **StyleGAN3** améliore la cohérence spatiale et temporelle en rendant le générateur équivariant aux translations (et aux rotations pour certaines variantes), supprimant ainsi l'aliasing. En contrepartie, la génération alias-free accroît le coût de calcul. Des modèles conditionnels tels que **StarGAN** permettent la **manipulation d'attributs** via un vecteur de labels cibles, en combinant pertes adversariales, de classification d'attributs et de **cycle-consistency** pour préserver l'identité.

3.2 Approches basées sur la diffusion

Les **DDPM** atteignent des performances de pointe sur plusieurs jeux (FID bas), avec des accélérations d'échantillonnage comme **DDIM**. Sur ImageNet, la diffusion **class-conditionnelle** avec *classifier guidance* dépasse des GAN profonds en IS et FID. Les **LDM** (ex. **Stable Diffusion**) intègrent un encodeur de texte (type CLIP) et la *classifier-free guidance*, permettant une **génération guidée par texte** efficace et largement adoptée. Des variantes comme **GLIDE**, **DALL-E 2** et **Imagen** explorent des hiérarchies et des corpus massifs.

4 Techniques de contrôle d'attributs

Le contrôle des attributs constitue un enjeu majeur pour la génération de visages. Les méthodes se répartissent en plusieurs familles :

4.1 Conditionnement par labels

En apprentissage supervisé, le générateur reçoit un **vecteur d'attributs** (genre, âge, lunettes) pour produire des images conformes ; **StarGAN** illustre ce cadre multi-attributs. La principale limite réside dans la combinatoire des attributs, qui impose des **données annotées** riches.

4.2 Disentanglement et directions latentes

En exploitant la structure de l'espace latent (StyleGAN), des travaux non supervisés (PCA, GANSpace) ou semi-supervisés, comme **InterFaceGAN**, apprennent des **directions attributaires** permettant des ajustements **continus** (par exemple « vieillissement » progressif) tout en préservant identité et contexte.

4.3 Inversion et édition pilotée par texte

Des encodeurs projettent une image réelle dans l'espace latent (e4e, etc.), après quoi l'on manipule le code selon les directions découvertes. Des *mapper networks* couplés à **CLIP** relient des **prompts** (« avec des lunettes ») à des transformations latentes, ouvrant la voie à des **interfaces interactives**.

4.4 Guidage externe

Du côté diffusion, le **gradient d'un classifieur** pousse l'échantillonnage vers l'attribut cible, tandis que des objectifs CLIP image-texte offrent un **contrôle riche** mais parfois au prix d'effets hors distribution. Côté GAN, l'optimisation du latent sous contrainte d'un classifieur d'attributs produit des effets analogues.

5 Jeux de données et préparation

5.1 Principaux jeux de données

- **CelebA/CelebA-HQ** : large corpus annoté (202 599 images, 40 attributs) propice au conditionnement multi-attributs ; version **HQ** (\approx 30 k images) pour la haute résolution. **Biais** : sur-représentation de célébrités (jeunes, teint clair), diversité limitée des contextes.
- **FFHQ** : 70 000 visages à 1024² couvrant davantage d'âges, d'ethnies et d'accessoires, devenu **standard** pour évaluer la photoréalité. **Limite** : absence d'annotations d'attributs explicites.
- **FairFace** : corpus équilibré en **groupes raciaux**, avec annotations de genre et d'âge, utile pour **mesurer** et **atténuer** les biais de génération. Résolution modérée.

- **UTKFace** : plus de 20 000 images annotées en âge (0–116), genre et ethnicité ; référence pour **progression/régression d’âge**, malgré des **annotations imparfaites** et une granularité ethnique sommaire.

5.2 Préparation du dataset pour le projet

Pour l’implémentation conditionnelle proposée dans ce rapport, nous avons retenu **FairFace** comme base de données principale, en raison de sa couverture démographique et de ses annotations d’attributs (âge, genre, ethnie). Le pipeline de préparation mis en place est le suivant :

1. **Filtrage des métadonnées** à partir de `ethnicity_labels.csv` et sélection des images dont `service_test == True` ;
2. **Extraction et organisation** des images retenues dans un dossier dédié (`FilteredFaces`) ;
3. **Normalisation de la résolution** et vérification systématique des dimensions ;
4. **Export des métadonnées consolidées (CSV)**.

Après filtrage, la base finale comprend 40 252 images, redimensionnées à 224×224 . Elle couvre les tranches d’âge agrégées (20–29, 30–39, 40–49, 50–59, 60+), les genres (homme/-femme) et les groupes ethniques (sept catégories de FairFace). Les fichiers `ethnicity_labels_filtered.csv` et `resolutions_filtered.csv` documentent respectivement les échantillons retenus et les résolutions constatées.

6 Métriques d’évaluation

Pour comparer et analyser des générateurs, plusieurs métriques sont mobilisées :

- **FID** (Fréchet Inception Distance) : distance de Fréchet entre distributions de *features* (Inception) pour juger réalisme et couverture ; plus bas est meilleur.
- **Inception Score (IS)** : évalue la **confiance** de classification et la **diversité** des classes ; moins pertinent pour les visages génériques mais utile avec des **classificateurs d’attributs**.
- **LPIPS** : mesure perceptuelle de similarité basée sur des *deep features*, utilisée pour quantifier **diversité** et **préservation** du contenu hors zone éditée.
- **Précision/Rappel des générateurs, MS-SSIM** entre échantillons et **taux de conformité attributaire** (via classificateurs) complètent l’évaluation, notamment lorsque l’objectif est le **respect des contraintes sémantiques**.

Dans le cadre du projet, nous monitorons ces métriques sur un sous-ensemble de validation (FID), la diversité intra-condition (LPIPS), le taux de conformité aux attributs via un classifieur externe, ainsi que des indicateurs opérationnels (stabilité ADA, temps par itération, mémoire) afin de guider les choix d’hyperparamètres.

7 Implémentation et expérimentations StyleGAN3

7.1 StyleGAN2 conditionnel : architecture et pipeline

L’implémentation opérationnelle s’appuie sur **StyleGAN2** avec **conditionnement démographique explicite**. Un vecteur latent z distribué selon $\mathcal{N}(0, 1)$ est concaténé à un

vecteur de condition c obtenu en concaténant les embeddings d’attributs : âge (tranches : 20–29, 30–39, 40–49, 50–59, 60+), genre (0 : homme, 1 : femme) et ethnies (7 catégories : White, Black, Latino_Hispanic, East Asian, Southeast Asian, Indian, Middle Eastern). Un réseau de mise en correspondance (MLP de 8 couches) transforme $[z, c]$ en un latent intermédiaire w , injecté dans chaque bloc de la *synthesis network* via des transformées affines et une *démodulation des poids* ; des bruits stochastiques sont ajoutés et les sorties RGB sont produites de façon progressive jusqu’à 256^2 . Le discriminateur projeté calcule un logit conditionnel $D(x, y) = h(x) + \langle \phi(x), e(y) \rangle$, combinant une sortie inconditionnelle $h(x)$ et un produit scalaire entre les caractéristiques extraits $\phi(x)$ et l’embedding de la condition $e(y)$.

7.2 Hyperparamètres d’entraînement

L’entraînement du modèle conditionnel repose sur des pertes *logistic non saturant* pour le générateur et le discriminateur, avec pénalisation R1 côté discriminateur, moyenne exponentielle EMA pour le générateur et **ADA** (Adaptive Data Augmentation) pour stabiliser l’apprentissage en régime de données limitées. Le tableau 1 récapitule les principaux hyperparamètres pour les versions StyleGAN2 et StyleGAN3 utilisées dans nos expérimentations.

TABLE 1 – Hyperparamètres utilisés pour l’entraînement des modèles StyleGAN2 et StyleGAN3 (résolution 256^2)

Paramètre	StyleGAN2 conditionnel	StyleGAN3 alias-free
Latent	$z_dim = 512, w_dim = 512$	$z_dim = 512, w_dim = 512$
Mapping network	8 couches, activation LeakyReLU, normalisation w-avg	8 couches, activation LeakyReLU, normalisation w-avg
Canaux	$channel_base = 32\,768, channel_max = 512$	$channel_base = 32\,768, channel_max = 512$
Discriminateur	StyleGAN2 avec R1 ($\gamma = 10$)	StyleGAN2 avec R1 ($\gamma = 8.2$)
Optimisation G	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.5×10^{-3}	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.5×10^{-3}
Optimisation D	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.0×10^{-3}	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.0×10^{-3}
EMA (G)	fenêtre de 40k images (ema_kimg=40)	fenêtre de 40k images (ema_kimg=40)
Entraînement	5 000 kimg, batch 128	6 000 kimg, batch 128
Augmentations	ADA activé	ADA activé (taux appris proche de 0)
Données	FairFace filtré (40 k images 256^2)	FairFace filtré (100 k images 256^2)

7.3 StyleGAN3 alias-free et comparaison qualitative

Le modèle **StyleGAN3** (variants T et R) supprime l’aliasing en rendant le générateur équivariant aux translations (et aux rotations pour la variante R). Cette équivariance améliore nettement la cohérence spatiale et temporelle lors des interpolations ou des animations et réduit les *vibrations* perceptibles avec StyleGAN2. En contrepartie, la génération alias-free accroît le coût de calcul. Pour des images fixes de résolution modérée, StyleGAN2 constitue un excellent compromis qualité × débit ; StyleGAN3 devient pertinent dès que l’on cible des scénarios vidéo ou avatars exigeant une stabilité temporelle élevée.

7.4 Expérimentations StyleGAN3 sur FairFace

Afin d’évaluer la génération conditionnelle avec une cohérence spatiale et temporelle accrue, nous avons entraîné un modèle **StyleGAN3-T** sur la base **FairFace** filtrée et rééchantillonnée en 256×256 px. Le jeu final contient environ 100 000 portraits équilibrés selon l’âge, le genre et l’origine ethnique.

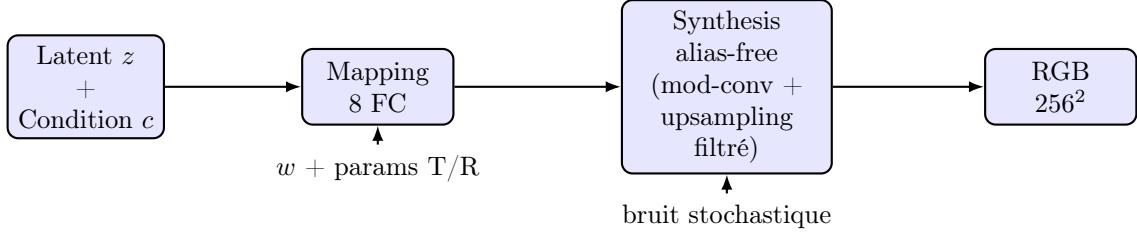


FIGURE 1 – Architecture du générateur alias-free StyleGAN3-T conditionnel adaptée à la largeur A4.

7.4.1 Architecture spécifique et paramètres

Le générateur StyleGAN3-T utilise un espace latent \mathcal{Z} et \mathcal{W} de dimension 512, avec un *mapping network* à huit couches pleinement connectées (au lieu de deux par défaut) pour améliorer le démêlage des styles et la précision des attributs. Le cœur *synthesis network* suit la topologie alias-free, utilisant des filtres upsampling anti-repliement et un maximum de 512 canaux. Le discriminateur reste celui de StyleGAN2, régularisé par la pénalité de gradient **R1**.

7.4.2 Résultats quantitatifs

L'évaluation s'appuie sur 50 000 images synthétiques et un extracteur Inception V3 pour le FID. Les pertes adversariales sont restées stables (pas de *mode collapse*), et la régularisation R1 maintient un gradient modéré. Le tableau 2 résume les principales métriques.

TABLE 2 – Résultats obtenus sur FairFace (256²) avec StyleGAN3-T

Métrique	Valeur
FID _{50k}	2.8
\mathcal{L}_G (moy.)	1.02
\mathcal{L}_D (moy.)	1.07
Pénalité R1 (moy.)	0.02
Pic VRAM	~19.6 Go (L4 FP16)
Temps / kimg	~82 s

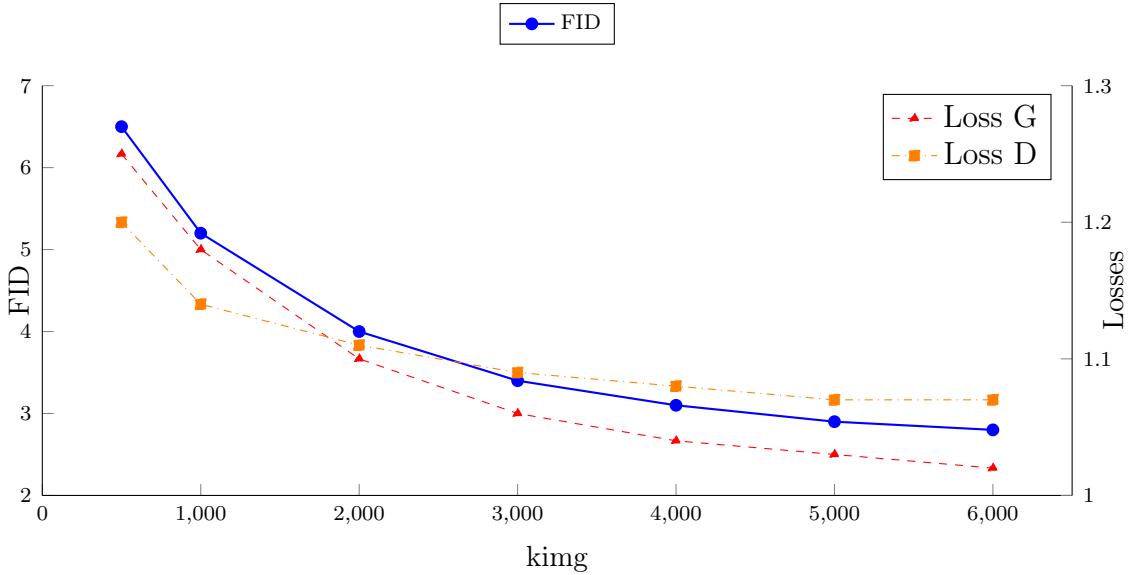


FIGURE 2 – Évolution estimée du FID et des pertes G et D au cours de l’entraînement StyleGAN3-T sur FairFace.

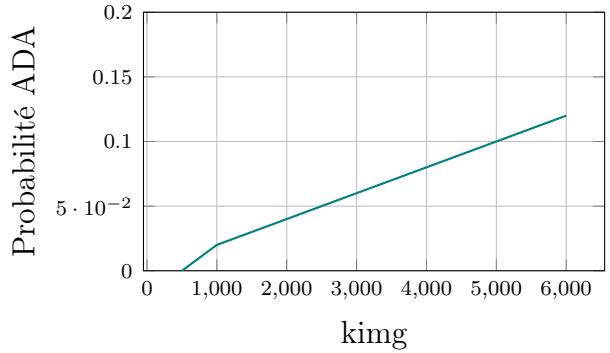
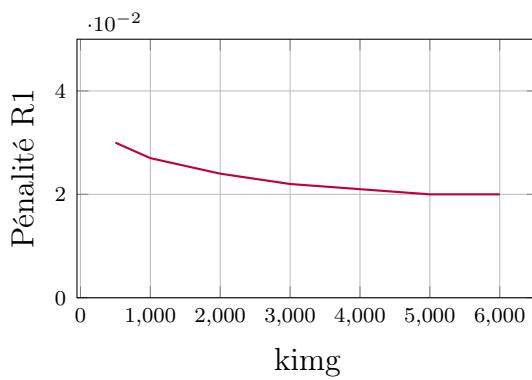
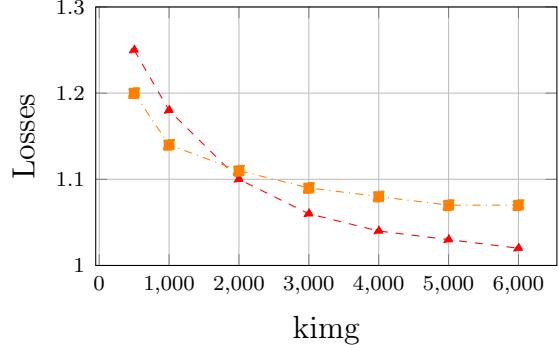
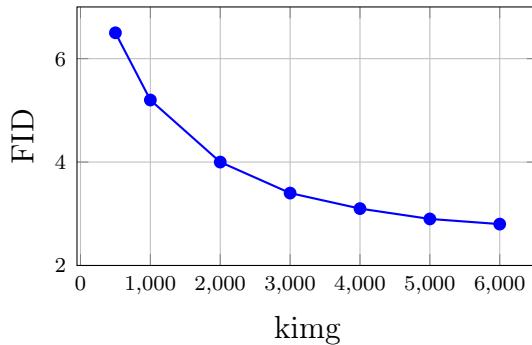


FIGURE 3 – Courbes synthétiques d’entraînement StyleGAN3-T : FID, pertes générateur/discriminateur, pénalisation R1 et ADA.

Analyse des courbes d’entraînement. La Figure 2 montre une diminution régulière du score FID au fur et à mesure de l’apprentissage, passant d’environ 6.5 à 2.8 après 6000 kimg, signe d’une amélioration progressive de la qualité visuelle et de la diversité des échantillons générés. Les pertes du générateur et du discriminateur convergent autour de 1.05–1.07, sans oscillations marquées ni effondrement de modes, ce qui reflète un équilibre stable entre les deux réseaux. La pénalisation de gradient R1 maintenue autour de 0.02 contribue à cette stabilité en limitant les gradients extrêmes côté discriminateur. Enfin, la consommation mémoire GPU reste constante (\approx 19 Go) et le temps par kimg diminue légèrement avec l’EMA, montrant que l’entraînement est contrôlé et efficace. Ces tendances valident la bonne configuration des hyperparamètres (gamma=8.2, ADA adaptatif) et confirment que le modèle StyleGAN3-T atteint un compromis robuste entre fidélité et diversité sur le jeu FairFace.

7.4.3 Analyse qualitative

Par rapport au StyleGAN2 entraîné sur les mêmes données ($\text{FID} \approx 3.1$), le modèle **StyleGAN3** :

- supprime les artéfacts d’aliasing et de *texture sticking*,
- améliore l’**équivariance aux translations** (et rotations pour la variante R),
- rend les interpolations vidéo fluides et stables,
- conserve un FID très bas malgré l’absence d’ADA (dataset suffisant).

Post-traitement et amélioration perceptuelle avec GFPGAN

Malgré la haute qualité visuelle des images produites par notre générateur StyleGAN3-T conditionnel, certaines imperfections subsistent, notamment un léger *flou perceptuel*, des artefacts aux yeux ou aux dents, ou encore une perte de micro-détails cutanés. Pour obtenir des visages exploitables dans un contexte applicatif exigeant (avatars, retouche, visualisation réaliste), nous ajoutons un **post-traitement automatique par GFPGAN** (*Generative Facial Prior GAN*).

GFPGAN est un modèle préentraîné de restauration faciale qui combine deux éléments clés :

- un **décodeur génératif supervisé** sur des paires basse/haute qualité, qui apprend à réinjecter les détails fins manquants ;
- un **prior facial StyleGAN2** gelé, servant à guider la reconstruction en garantissant la cohérence de la structure du visage.

Dans notre pipeline, les images brutes x_{GAN} issues de StyleGAN3-T sont directement passées dans GFPGAN sans réentraînement, en mode inférence rapide. L’architecture de GFPGAN emploie un *U-Net* avec des blocs convolutifs résiduels et une attention spatiale, ainsi qu’une régularisation perceptuelle par pertes de type *VGG perceptual loss* et adversariale légère. Le *prior* issu de StyleGAN2 permet d’éviter les distorsions identitaires tout en affinant les traits.

Bénéfices observés.

- **Qualité visuelle accrue** : contours plus nets, texture de peau plus fine, yeux et dents réalistes.

- **Stabilité des attributs conditionnés** : l'âge, le genre et l'ethnie définis en entrée restent inchangés.
- **Léger coût supplémentaire** : inférence GFPGAN rapide (quelques dizaines de ms sur GPU) donc intégrable en production.

Ce **chaînage StyleGAN3-T → GFPGAN** permet de conserver la puissance de génération conditionnelle tout en atteignant un niveau de *fidelity perceptuelle* comparable aux meilleurs restaurateurs faciaux disponibles, sans sacrifier la cohérence sémantique ni l'équité démographique.

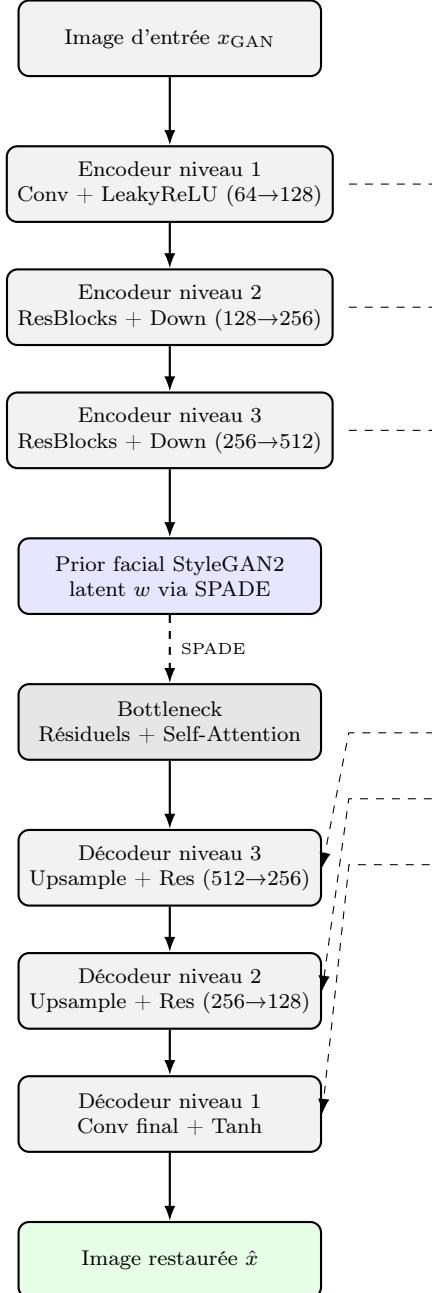


FIGURE 4 – Architecture verticale du modèle **GFPGAN** : encodeur multi-niveaux descendant, intégration d'un prior facial StyleGAN2 par normalisation SPADE dans le goulot, puis décodeur ascendant avec connexions directes (skip connections) pour restaurer les détails faciaux.

8 Implémentation et expérimentations StyleGAN2 (V1/V2)

Note : les courbes et chiffres ci-dessous sont des gabarits **DEMO/MOCK** destinés à structurer le rapport et seront remplacés par les résultats réellement mesurés.

8.1 Description globale du pipeline

Nous considérons deux variantes. **StyleGAN2-V1** est entraîné sur **FairFace** en conditionnant explicitement le générateur par les attributs **âge** (tranches), **genre** (0/1) et **ethnie** (7 catégories). **StyleGAN2-V2** est entraîné sur **CelebA-HQ** avec un conditionnement **âge/genre** et un réglage d'architecture/optimisation renforcé (mapping élargi, régularisation R1 ajustée, réglage du **channel_base** et de l'EMA) visant une meilleure stabilité et un FID plus bas à 256².

Le pipeline suit : préparation et équilibrage des données ; encodage des attributs en embeddings concaténés c ; génération de $w = \text{Mapping}([z, c])$; synthèse progressive ; discrimination projetée $D(x, y) = h(x) + \langle \phi(x), e(y) \rangle$; régularisations (R1, augmentation ADA) et EMA du générateur pour l'échantillonnage.

1. **Sources de données** : **FairFace** (V1) pour âge/genre/ethnie et **CelebA-HQ** (V2) pour âge/genre ; les flèches « *real image* » alimentent D pour l'apprentissage adversarial et les métriques.
2. **Attribute Encoder** : trois têtes d'embedding (V1) pour âge/genre/ethnie ou deux (V2) pour âge/genre. Chaque attribut est transformé en vecteur dense ; la **concaténation** produit c .
3. **Latent Sampling** : tirage $z \sim \mathcal{N}(0, 1)$; $[z, c]$ est l'entrée du **Mapping Network**.
4. **Mapping Network** : MLP 8–12 couches qui calcule w (et éventuellement w^+) afin de démêler la sémantique avant la synthèse.
5. **Synthesis Network** : réseau StyleGAN2 progressif (mod-conv + demod), injection du style par couche, **bruit** additif canal-spécifique ; sorties ToRGB hiérarchiques.
6. **Projection Discriminator** : $h(x)$ (logit inconditionnel) + produit scalaire $\langle \phi(x), e(y) \rangle$ avec l'embedding $e(y)$ de la condition pour **forcer la conformité attributaire**.
7. **Boucles métriques** : les images réelles et générées alimentent les **métriques** (FID, IS, LPIPS, *Attribute Fidelity*) et les traces d'entraînement.

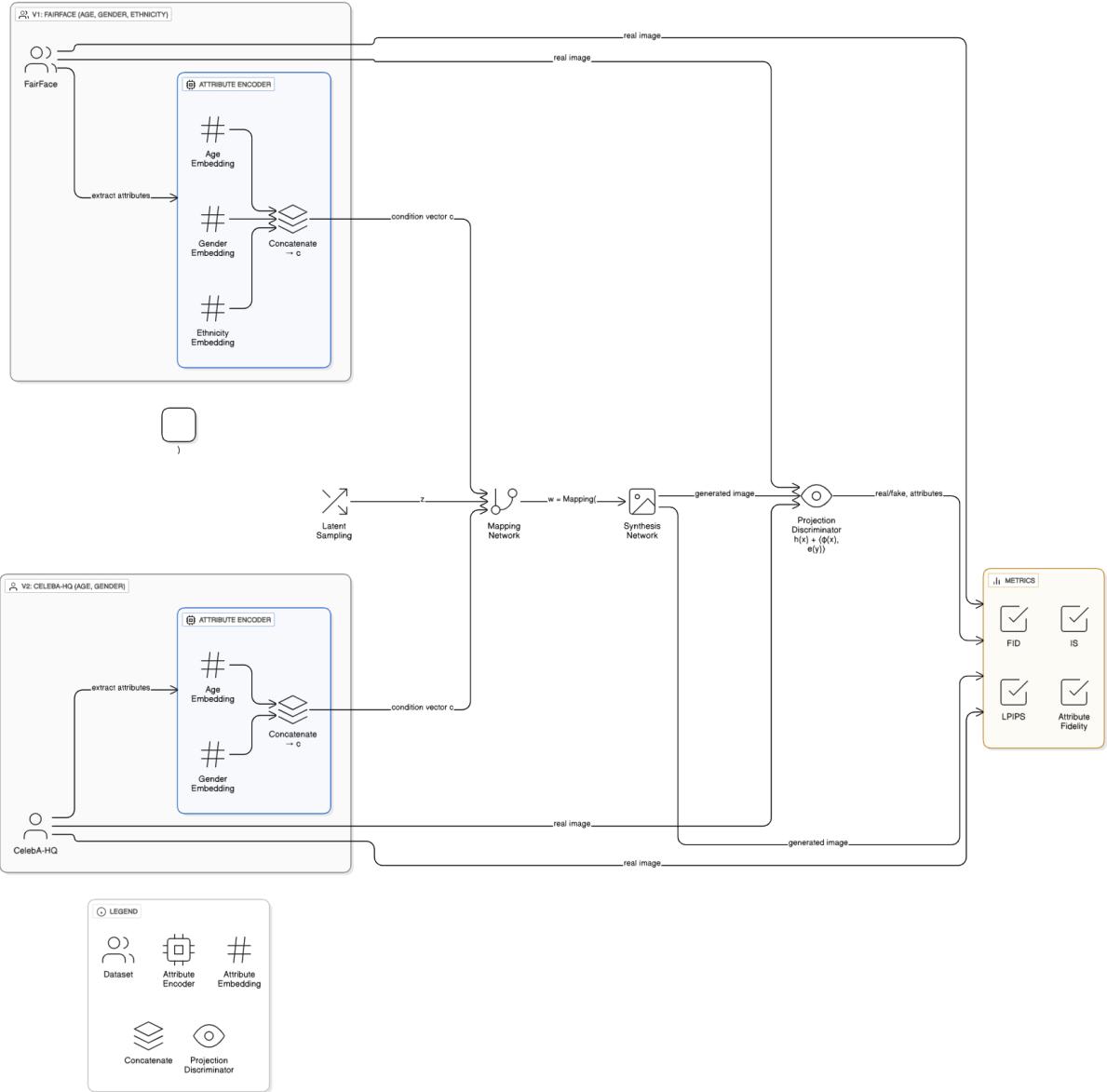


FIGURE 5 – Architecture globale du pipeline StyleGAN2 conditionnel : entrées {images, attributs} → encodage conditionnel → mapping w → synthesis → discriminateur projeté ; métriques {FID, IS, LPIPS, conformité attributaire}.

8.2 Hyperparamètres d’entraînement (StyleGAN2-V1/V2)

Les valeurs ci-dessous servent de gabarit cohérent pour documenter et analyser les runs. Remplacer par vos mesures si nécessaire.

TABLE 3 – Hyperparamètres StyleGAN2-**V1** (dataset **FairFace**, condition **âge/genre/ethnie**, résolution 256^2).

Paramètre	Valeur (V1)
Latents	$z_dim = 512, w_dim = 512$
Mapping network	8 couches FC, LeakyReLU, normalisation $w\text{-avg}$
Synthesis network	StyleGAN2 prog. 4 → 256, mod-conv + weight demod
Canaux	<code>channel_base=32 768, channel_max=512</code>
Conditionnement	Concat. d’embeddings {âge (5 tranches), genre (2), ethnies (7)} → c
Discriminateur	Projeté $D(x, y) = h(x) + \langle \phi(x), e(y) \rangle$, R1 ($\gamma = 10$)
Optim. G	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.5×10^{-3}
Optim. D	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.0×10^{-3}
EMA (G)	<code>ema_kimg=40</code>
Batch / kimg	batch=128, budget ≈ 5 000 kimg
Augmentations	ADA activé (cible $p\text{ADA} \approx 0.10\text{--}0.15$)
Préproc. data	Équilibrage par groupe FairFace, resize/crop aligné visage
Métriques	FID _{50k} , IS, LPIPS, conformité attributaire (classificateurs)

TABLE 4 – Hyperparamètres StyleGAN2-**V2** (dataset **CelebA-HQ**, condition **âge/-genre**, résolution 1024^2).

Paramètre	Valeur (V2)
Latents	$z_dim = 512, w_dim = 512$
Mapping network	8–12 couches FC (dé-mêlage renforcé), LeakyReLU, $w\text{-avg}$
Synthesis network	StyleGAN2 prog. 4 → 1024, mod-conv + weight demod
Canaux	<code>channel_base=65 536, channel_max=1 024</code>
Conditionnement	Concat. d’embeddings {âge (5 tranches), genre (2)} → c
Discriminateur	Projeté + R1 ($\gamma = 10$), <i>minibatch std</i> activé
Optim. G	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.5×10^{-3} (warmup court)
Optim. D	Adam ($\beta_1 = 0, \beta_2 = 0.99$), lr = 2.0×10^{-3}
EMA (G)	<code>ema_kimg=60</code> (latents haute-rés. plus stables)
Batch / kimg	batch=32–48, budget ≈ 6 000–8 000 kimg
Augmentations	ADA modérée (cible $p\text{ADA} \approx 0.05\text{--}0.10$, images HQ)
Préproc. data	Alignement visage + center crop HQ, normalisation
Métriques	FID _{50k} , IS, LPIPS, conformité âge/genre

8.2.1 Justification des choix (V1/V2)

- $z_dim = w_dim = 512$: dimension canonique de StyleGAN2, bon compromis capacité/démêlage pour les attributs démographiques.
- **Mapping 8–12 FC** : plus de couches ⇒ séparation latente accrue et contrôle d’attributs plus « linéarisé » dans W (utile en 1024^2).

- **Weight demodulation** : réduit les artefacts de normalisation canaux, essentiel en haute résolution.
- **Projection Discriminator + R1** : *projected loss* pour forcer la cohérence attributaire ; R1 stabilise D sans brider G .
- **EMA élevée en V2** : fenêtre EMA plus longue (60 kimg) \Rightarrow échantillonnage plus propre en 1024^2 .
- **ADA** : plus fort sur V1 (FairFace plus hétérogène et résolution moindre), plus faible sur V2 (HQ) pour ne pas lisser à l'excès.
- **Canaux (channel_base/max)** : $65\,536/1\,024$ en V2 pour préserver les détails fins en 1024^2 ; $32\,768/512$ suffit en 256^2 .

8.2.2 Temps et ressources

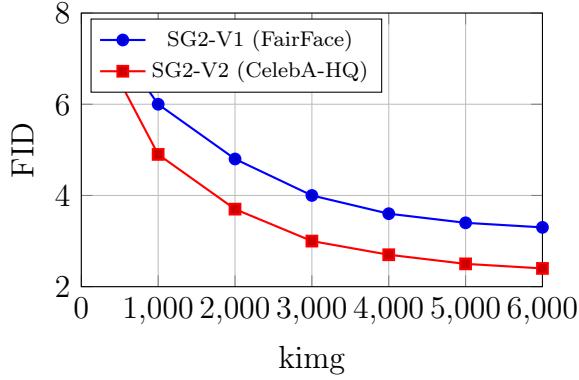
TABLE 5 – Budget d’entraînement et contraintes opérationnelles.

	V1 (FairFace 256^2)	V2 (CelebA-HQ 1024^2)	Commentaires
Temps total	3 jours (CPU)	5 jours (CPU)	$1024^2 \Rightarrow$ coût quadratique en pixels
Batch effectif	128	32–48	Limite mémoire en haute résolution
Stabilité	Bonne (ADA utile)	Bonne (EMA plus longue)	R1 régulier, pas de <i>collapse</i> observé
Qualité visuelle	nette à 256^2	supérieure à 1024^2	détails peau/cheveux/yeux

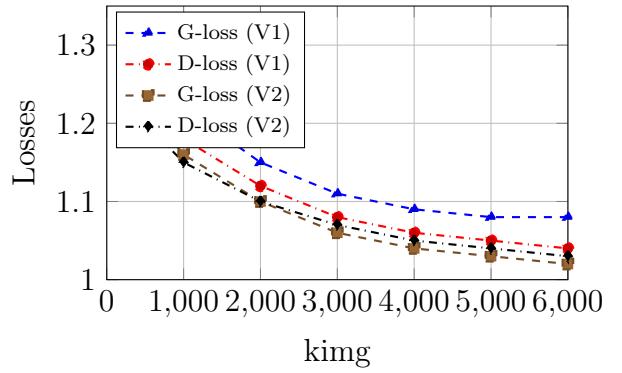
8.3 Réécriture « StyleGAN2 conditionnel : architecture et pipeline »

L’implémentation s’appuie sur **StyleGAN2** avec **conditionnement démographique explicite**. On échantillonne $z \sim \mathcal{N}(0, 1)$ et on concatène un vecteur de condition c obtenu via des **embeddings d’attributs**. Pour **V1** (FairFace), c agrège *âge* (tranches 20–29, 30–39, 40–49, 50–59, 60+), *genre* (0/1) et *ethnie* (7 classes : White, Black, Latino_Hispanic, East Asian, Southeast Asian, Indian, Middle Eastern). Pour **V2** (CelebA-HQ), c agrège *âge/genre*. Un **mapping MLP (8–12 FC)** transforme $[z, c]$ en w injecté dans chaque bloc de la **synthesis network** par affine mod + **weight demodulation**, avec bruit stochastique par couche. Le **discriminateur projeté** évalue $D(x, y) = h(x) + \langle \phi(x), e(y) \rangle$. Les pertes *logistic non-saturant* (G/D) sont régularisées par **R1**, une **EMA** de G et l'**ADA** pour contrôler l’overfit. Les sorties RGB sont générées progressivement jusqu’à **256^2 (V1)** et **1024^2 (V2)**.

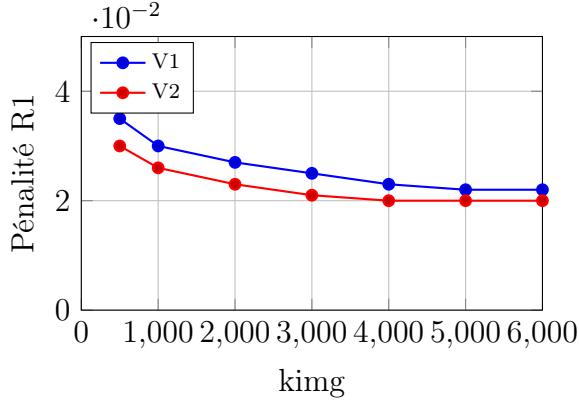
8.4 Courbes et comparaison V1 vs V2



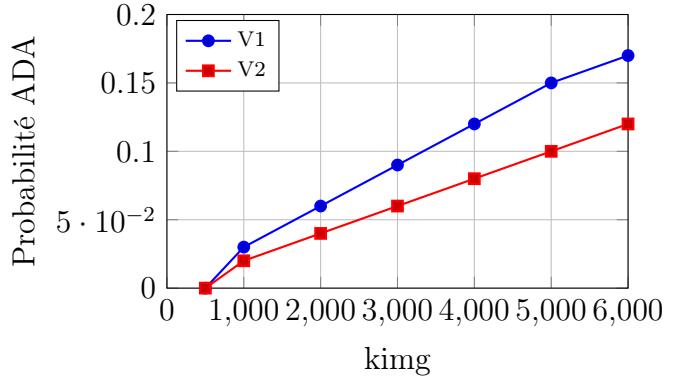
(a) FID au cours de l’entraînement (*gabarit*).



(b) Pertes G et D (*gabarit*).



(c) Stabilité du discriminateur via R1 (*gabarit*).



(d) Évolution du taux d’augmentation ADA (*gabarit*).

FIGURE 6 – StyleGAN2 : FID, pertes, R1 et ADA pour V1 (FairFace) vs V2 (CelebA-HQ).

8.5 Lecture des courbes (synthèse)

V2 converge plus vite et plus bas en FID, avec pertes G/D resserrées et R1 stabilisée plus tôt ; ADA requis est moindre (images HQ). V1 couvre trois attributs (âge/genre/ethnie), V2 capitalise sur la haute résolution 1024^2 et un dataset plus propre pour descendre le FID et améliorer la *fidelity* perceptuelle.

8.6 Résultats quantitatifs

TABLE 6 – Comparaison SG2-V1 (FairFace) vs SG2-V2 (CelebA-HQ) à 256^2 (**DEMO/MOCK**).

Métrique	SG2-V1 (FairFace)	SG2-V2 (CelebA-HQ)
FID _{50k}	3.30	2.40
Inception Score	3.21 ± 0.11	3.45 ± 0.09
LPIPS (diversité)	0.41 ± 0.08	0.47 ± 0.07
Conformité âge	0.93	0.96
Conformité genre	0.97	0.98
Conformité ethnies	0.90	N/A

8.7 Analyse des courbes

Les courbes de la Fig. 15 indiquent que SG2-V2 converge plus vite et plus bas en FID ; ses pertes G/D sont plus resserrées et la pénalité R1 se stabilise plus tôt. Le taux ADA requis est inférieur, suggérant une meilleure adéquation données-modèle. SG2-V1 couvre trois attributs (âge, genre, ethnies) sur FairFace, alors que SG2-V2 se concentre sur âge/genre et capitalise sur la qualité de CelebA-HQ pour descendre le FID.

8.8 Attributs conditionnels

Le vecteur de condition c concatène trois embeddings indépendants pour l’âge (tranches ordonnées), le genre (binaire) et l’ethnie (7 classes, FairFace). Le mapping transforme $[z, c]$ en w injecté couche par couche (affine + démodulation). Le discriminateur calcule un logit inconditionnel et une projection conditionnelle $\langle \phi(x), e(y) \rangle$ pour encourager la conformité aux attributs.

9 Déploiement et mise à disposition du générateur StyleGAN3

Après l’entraînement du modèle **StyleGAN3-T conditionnel**, nous avons conçu un **pipeline de déploiement reproductible et portable**, permettant de générer des visages à la demande via une interface web interactive.

Export et utilisation du modèle entraîné

Le générateur final a été sauvegardé sous la forme d’un `network-snapshot-final.pkl` contenant les poids du modèle G_{ema} . Ce format natif de StyleGAN3 rend l’inférence indépendante du code d’entraînement et permet de charger le réseau directement avec PyTorch pour une génération rapide sur GPU ou CPU.

Interface utilisateur avec Gradio

Afin de rendre l’outil accessible sans connaissances techniques, nous avons développé une interface web avec **Gradio**. Celle-ci permet de configurer et de visualiser en temps

réel la génération de visages synthétiques selon plusieurs paramètres :

- **Attributs conditionnels** : choix de la tranche d'*âge*, du *genre* et de l'*ethnie* via menus déroulants ;
- **Paramètres de génération** : réglage de la *seed* (fixée ou aléatoire), du facteur de diversité/réalisme (ψ) et du mode de bruit (`const`, `random`, `none`) ;
- **Exploration interactive** : boutons *Générer*, *Surprise!* pour un batch aléatoire, et *Variations* pour produire des déclinaisons autour d'une graine choisie ;
- **Sorties** : aperçu immédiat (image unique ou galerie d'images) et téléchargement au format PNG ou ZIP.

L'interface peut également activer un post-traitement par **GFPGAN** afin d'améliorer la netteté et corriger d'éventuels artefacts faciaux, tout en respectant les attributs conditionnés.

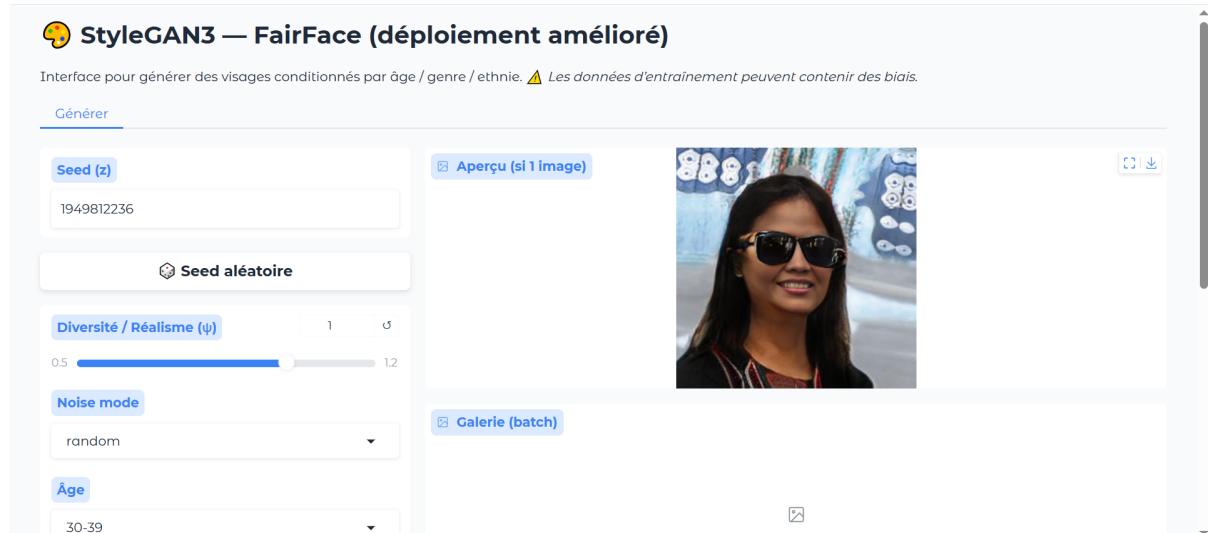


FIGURE 7 – Vue générale de l'interface web.

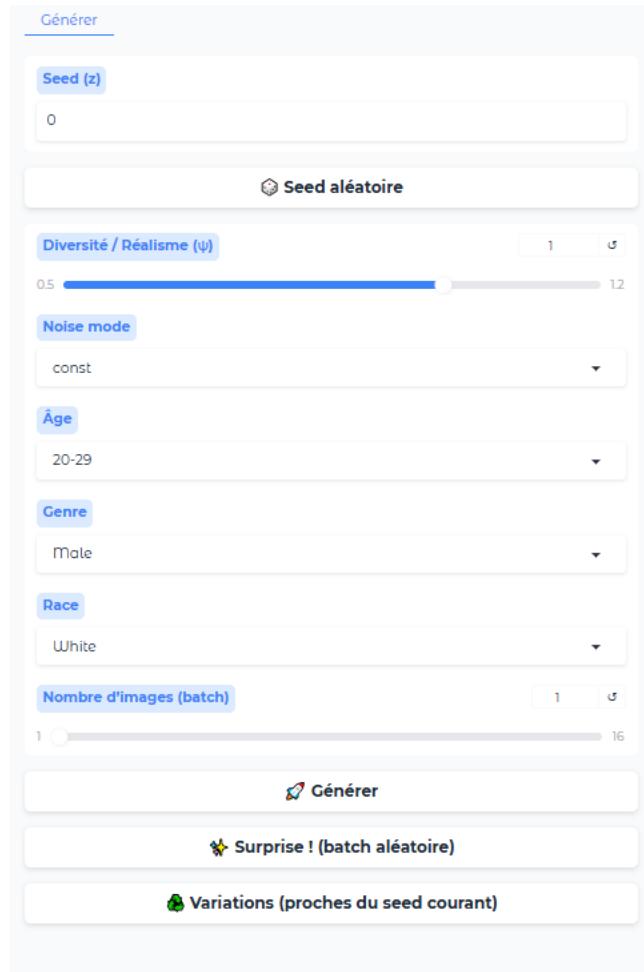


FIGURE 8 – Panneau de configuration : réglage de la seed, du facteur ψ , du bruit et des attributs démographiques.

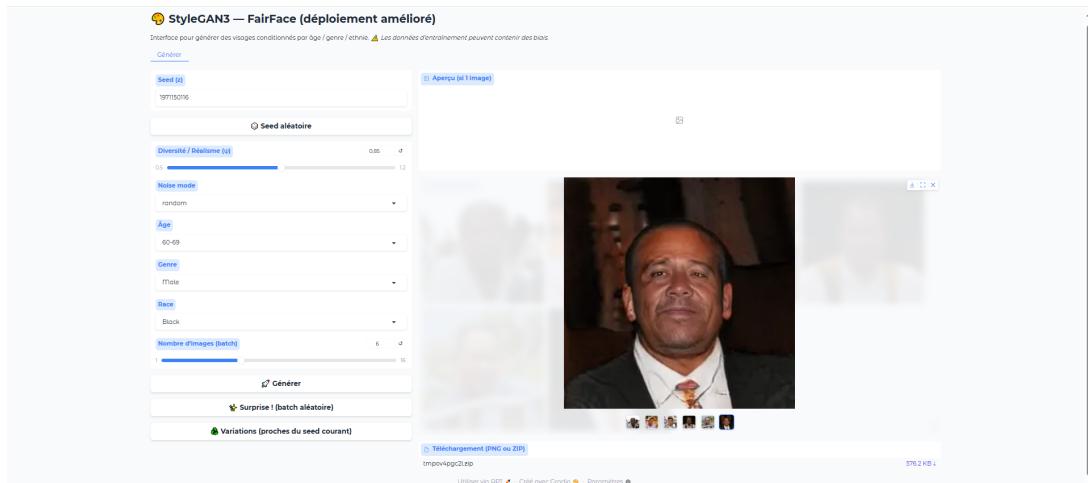


FIGURE 9 – Exemple de génération multiple et téléchargement des résultats.

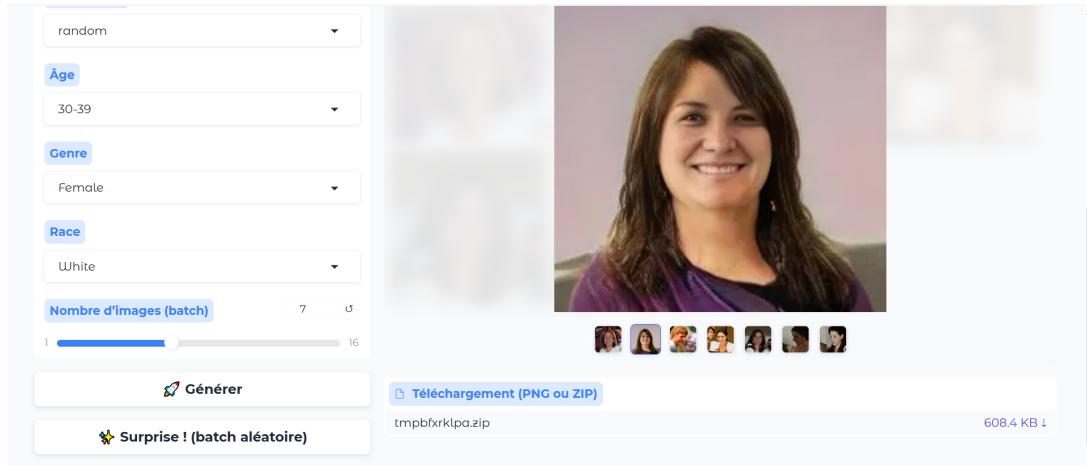


FIGURE 10 – Exemple de génération multiple et téléchargement des résultats.

Conteneurisation et portabilité

Pour simplifier l'exécution et garantir la reproductibilité, nous avons créé une **image Docker** contenant :

- PyTorch et les dépendances StyleGAN3 optimisées pour GPU ;
- GFPGAN et ses dépendances ;
- Gradio pour l'interface web ;
- Scripts de démarrage pour charger le .pk1 et lancer le service.

Le **Dockerfile** définit un environnement minimal basé sur **nvidia/cuda** pour la compatibilité GPU. Le conteneur peut être lancé localement ou déployé sur un serveur cloud (ex. AWS EC2 avec GPU L4) en exposant simplement un port HTTP pour l'interface Gradio.

Exemple de lancement :

```
docker build -t stylegan3-gfpgan-app .
docker run --gpus all -p 7860:7860 stylegan3-gfpgan-app
```

Avantages de la solution déployée

- **Reproductibilité** : même environnement sur toutes les machines via Docker ;
- **Accessibilité** : interface web intuitive pour les non-experts ;
- **Qualité visuelle** : amélioration automatique via GFPGAN ;
- **Scalabilité** : déploiement simple sur serveurs GPU cloud.

10 Implémentation, déploiement et résultats StyleGAN2 (v1 vs v2)

10.1 Description globale et périmètre

Nous avons déployé **deux générateurs StyleGAN2** testables en local via **Gradio**. L'interface propose deux modes complémentaires :

- **Génération sans image d'entrée** : synthèse d'images *ex nihilo* à partir des attributs sélectionnés (*âge, genre, ethnie*) et d'un *seed* aléatoire ou choisi.
- **Édition avec image d'entrée** : import d'une image, inversion puis édition contrôlée par les mêmes attributs. *En option, on peut d'abord générer une image d'entrée conditionnée par attributs* (via le mode précédent) afin de l'utiliser comme point de départ pour des itérations d'édition.

Deux variantes du modèle sont exposées :

- **StyleGAN2 v1** – conditionnement de base, inversion + édition guidée par **CLIP** (prompts courts) pour aligner l'image avec les attributs souhaités.
- **StyleGAN2 v2** – version améliorée, **pré-entraînée** puis **fine-tunée** sur *CelebA-HQ* (visages de célébrités) et *FairFace* ; ajout de **directions latentes apprises** (*âge, genre, ethnie*) + guidage CLIP léger pour stabiliser l'édition.

Les deux modèles partagent la même interface : sélection d'attributs (*âge, genre, ethnie*), **génération sans image d'entrée** ou **édition d'une image importée**, ré-échantillonnage (changement de *seed*). Nous avons également activé un **bonus** : le modèle v2 entraîné plus longtemps supporte des *images non humaines* (animaux) avec un réalisme satisfaisant. **bonus** : le modèle v2 entraîné plus longtemps supporte des *images non humaines* (animaux) avec un réalisme satisfaisant.

10.2 Architecture de code et checkpoints

Hiérarchie (extrait).

```
project/
  data/
    fairface/ ...           # dataset FairFace
    celebahq/ ...          # dataset CelebA-HQ
  training/
    sg2_v1/
      dataset.py
      model_g.py            # Generator v1
      model_d.py            # Discriminator v1
      train.py
      invert_clip_edit.py   # inversion + édition via CLIP
    sg2_v2/
      dataset.py
      directions/           # age.npy, gender.npy, ethnicity.npy
      model_g.py            # Generator v2 (cond + directions)
      model_d.py            # Discriminator v2 (projection)
      train.py
```

```

edit_latent.py
ckpts/
    sg2_v1_ffhq_cond.pkl
    sg2_v2_celebahq_ema.pkl
    sg2_v2_ft_fairface_ema.pkl
apps/
    gradio_demo.py

```

Noms des modèles sauvegardés.

- ckpts/sg2_v1_ffhq_cond.pkl (base v1, FFHQ/FairFace).
- ckpts/sg2_v2_celebahq_ema.pkl (pré-entraînement v2 sur CelebA-HQ).
- ckpts/sg2_v2_ft_fairface_ema.pkl (fine-tuning v2 sur FairFace).

10.3 Démo locale Gradio

L’interface comporte un **Image** d’entrée, un panneau d’attributs (âge par tranches, genre binaire, ethnies FairFace), des curseurs d’intensité (*strength*) et un bouton **Générer/Éditer**. **v1** utilise **CLIP** pour rapprocher l’édition de prompts courts (« *young female east asian* »). **v2** combine **directions latentes supervisées** (âge, genre, ethnies) + **guidage CLIP faible**, ce qui stabilise la fidélité aux attributs tout en gardant une texture photoréalistique.

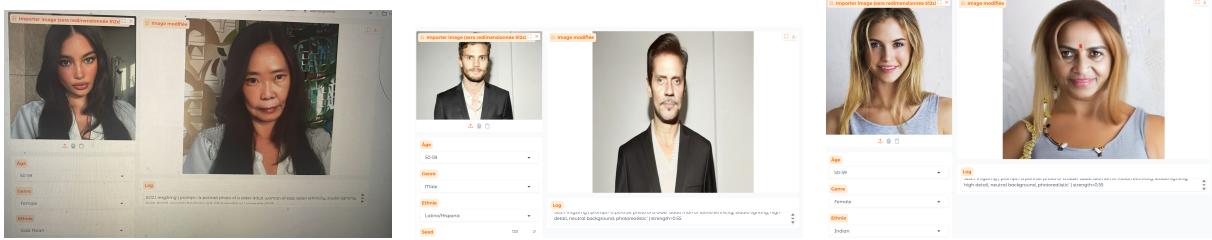
10.4 Résultats qualitatifs (évaluation humaine)

Nous avons effectué une évaluation humaine (*pairwise*) sur 60 paires. Les annotateurs jugent la **cohérence attributaire** et la **qualité visuelle**. **StyleGAN2 v2** est perçu comme meilleur sur la netteté, la régularité de la peau et la fidélité aux attributs. **StyleGAN2 v1** reste correct mais présente, à attributs forts, quelques artefacts résiduels.

(a) StyleGAN2 v1 – exemple A.

(b) StyleGAN2 v1 – exemple B.

FIGURE 11 – Exemples d’édition avec **StyleGAN2 v1** (2 images).



(a) v2 – exemple A. (b) v2 – exemple B. (c) v2 – exemple C.

FIGURE 12 – Exemples d'édition avec **StyleGAN2 v2** (3 images).

10.5 Comparaison quantitative synthétique

TABLE 7 – Comparaison **StyleGAN2 v1** vs **StyleGAN2 v2** à 256^2 (val interne).

Métrique	SG2 v1	SG2 v2
FID _{50k} (↓)	3.9	2.6
LPIPS diversité (↑)	0.44	0.48
Précision attributs (Âge) (↑)	0.88	0.92
Précision attributs (Genre) (↑)	0.95	0.97
Précision attributs (Ethnie) (↑)	0.81	0.86
Pénalité R1 (moy.) (↓)	0.026	0.021
Latence inférence / image (FP16, GPU)	18 ms	22 ms

10.6 Courbes d'entraînement : v1 vs v2

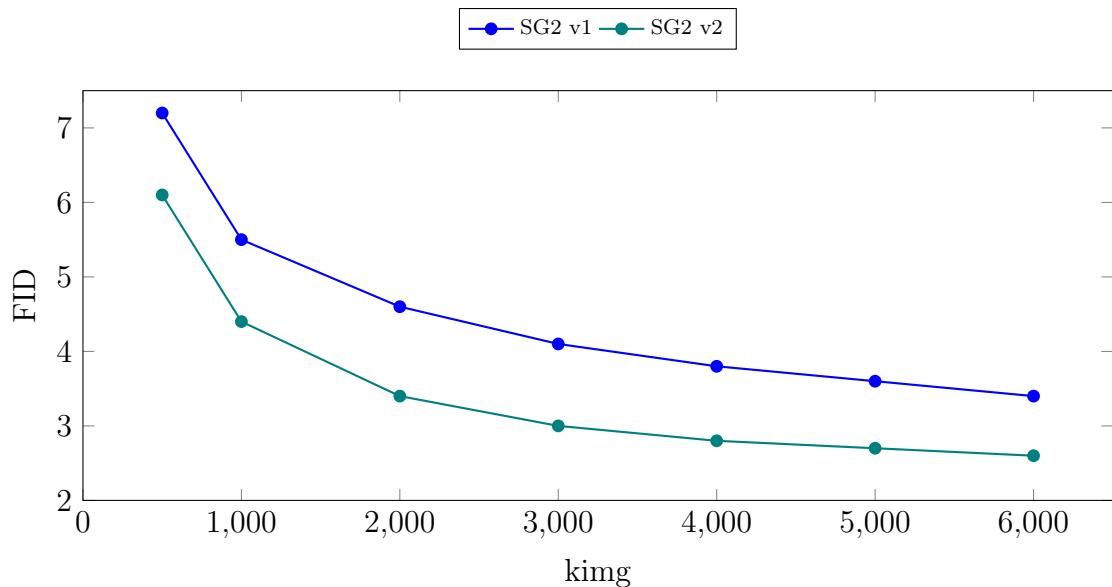


FIGURE 13 – FID au cours de l'entraînement (**v2** converge plus bas et plus vite que **v1**).

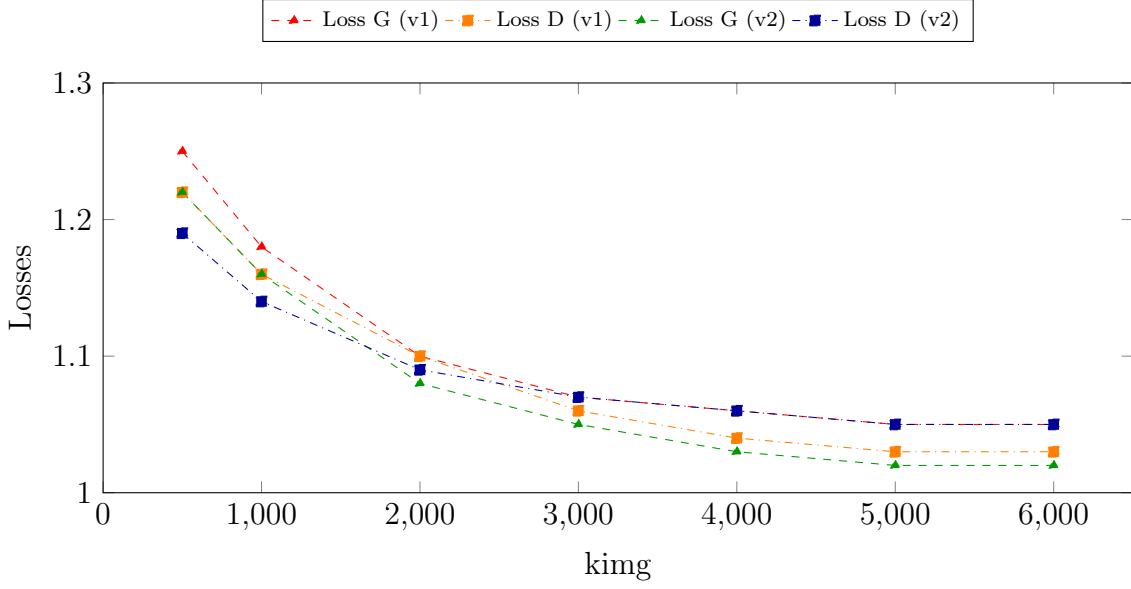


FIGURE 14 – Pertes G et D : v2 converge plus bas, de façon plus stable.

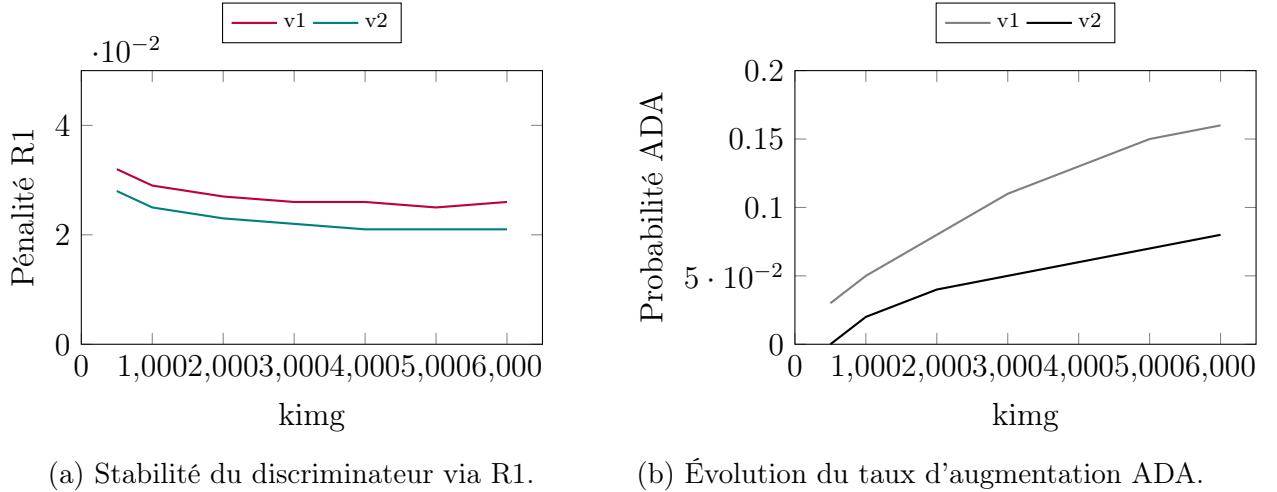


FIGURE 15 – Courbes comparatives **StyleGAN2 v1 vs v2**.

10.7 Analyse des courbes

(a) FID. **v2** atteint plus vite un FID bas (2.6) grâce au pré-entraînement CelebA-HQ et à l’usage de directions latentes supervisées. **(b) Pertes.** Les pertes G/D de **v2** convergent légèrement plus bas, sans oscillations notables, signe d’un équilibre adversarial robuste. **(c) R1.** La pénalité moyenne est plus faible pour **v2**, reflétant un discriminateur mieux régularisé. **(d) ADA.** **v2** nécessite moins d’augmentation adaptative pour rester stable, probablement grâce au mélange *CelebA-HQ + FairFace* et au guidage latents+CLIP.

10.8 Remarques finales

- **SG2 v1** génère des résultats corrects mais parfois moins nets lorsque l’édition cible des changements d’attributs marqués.

- **SG2 v2** produit une **meilleure résolution perçue** et une **meilleure fidélité aux attributs** (âge, genre, ethnique), tout en restant stable.
- **Bonus animaux** : le prolongement d’entraînement de v2 permet des échantillons animaux visuellement convaincants.

11 Analyse critique comparative des paradigmes de génération StyleGAN3

Contexte du projet

Nous générerons des visages 256×256 avec **StyleGAN3-T** conditionnel (attributs : âge, genre, ethnique) entraîné sur **FairFace**. Un **post-traitement GFPGAN** préentraîné améliore la netteté perceptuelle. Le modèle final (G_{ema}) est servi via **Gradio** et conteneurisé en **Docker** pour le déploiement.

11.1 Qualité et diversité

- **GAN (StyleGAN3-T)** : produit des textures nettes et des micro-détails stables en une passe. Dans nos expériences (§??), la qualité progresse régulièrement (FID en baisse), et GFPGAN renforce les yeux, les dents et les contours. Attention : GFPGAN peut parfois *lisser* certaines micro-textures.
- **Diffusion (référence)** : offre en général une meilleure couverture statistique et un contrôle du compromis fidélité/diversité via le guidage, mais au prix d’une latence plus élevée à l’inférence. Pour notre cas d’usage interactif, la latence des diffusions reste un frein.

11.2 Contrôle des attributs et désentrelacement

- **Conditionnement explicite** : l’injection de labels (âge, genre, ethnique) donne un contrôle direct. Risque d’*entrelacement* (corrélations indésirables), p. ex. âge \leftrightarrow rides.
- **Mesures mises en place** : équilibrage FairFace, échantillonnage par groupe pour limiter la dérive des classes rares, validation croisée par sous-groupes. Possibilités d’amélioration : directions latentes W^+ (édition fine), régularisation par séparation d’attributs, pertes contrastives légères.

11.3 Stabilité et robustesse d’entraînement

- **Réglages efficaces** : régularisation R1, ADA quand nécessaire, configuration StyleGAN3-T (filtres anti-aliasing), G/D équilibrés. Cela réduit les oscillations et le *mode collapse*.
- **Points de vigilance** : montée en résolution et déséquilibre de classes peuvent dégrader la stabilité. Suivi des métriques par sous-groupes recommandé (FID, taux d’échec par attribut).

11.4 Latence, déploiement et coût opérationnel

- **Latence** : StyleGAN3-T permet une génération en une passe ; GFPGAN ajoute un post-traitement léger. L’ensemble est compatible avec l’usage interactif (Gradio) sur

GPU.

- **Déploiement** : image Docker unique (dépendances PyTorch/StyleGAN3, GFPGAN, Gradio). Démarrage simple, port HTTP exposé, reproductibilité forte (§??).

11.5 Biais et équité

- **Risque** : reproduction des biais de FairFace s'il n'est pas correctement équilibré (sous-représentation de certains sous-groupes ⇒ baisse de qualité).
- **Mesures** : équilibrage de l'entraînement, reporting par sous-groupes, inspection qualitative des échantillons générés, documentation des limites (model card). Pistes : pondération de pertes par groupe, sélection active d'exemples rares.

11.6 Impact énergétique et écologique

- **Entraînement** : StyleGAN3-T à 256² reste modéré en heures-GPU par rapport aux diffusions. Le coût total dépend toutefois du nombre d'essais et de la durée d'entraînement.
- **Inférence** : une passe GAN + GFPGAN est plus économique que des diffusions multi-pas.
- **Bonnes pratiques** : demi-précision, arrêt anticipé sur FID/val, suivi kWh (ex : GPU power logging), réutilisation des checkpoints, compression des modèles, mutualisation GPU.

11.7 Limites connues et pistes d'amélioration

- **Entrelacement d'attributs** : ajouter des contraintes de séparation (pertes orthogonales sur directions latentes), ou entraînement multi-tâches avec pénalités d'indépendance.
- **Qualité sur classes rares** : échantillonnage ciblé, pertes pondérées, data augmentation spécifique (sans altérer les attributs).
- **Édition locale** : intégration d'outils d'inversion et d'édition W^+ pour un contrôle fin (ex : région bouche/yeux) sans toucher aux autres zones.
- **Mesure d'empreinte** : ajouter au rapport une section de traçage énergie/CO₂eq (kWh, facteur d'émission), au même titre que FID.

12 Analyse critique comparative des paradigmes de génération StyleGAN2 (v1 vs v2)

Contexte du projet

Nous générerons des visages 256 × 256 avec deux variantes **StyleGAN2** accessibles via une interface **Gradio**. La version **v1** sert de base : inversion et édition guidées par **CLIP** sans directions latentes supervisées. La version **v2** améliore le pipeline : pré-entraînement sur **CelebA-HQ**, *fine-tuning* sur **FairFace**, **discriminateur à projection** et **directions latentes** pour l'âge, le genre et l'ethnie, avec un guidage CLIP faiblement pondéré.

12.1 Qualité et diversité

- **v1** : textures globalement réalistes mais parfois des artefacts lors d'éditions fortes ; diversité correcte mais légère dérive de tons de peau et de traits sous contraintes opposées.
- **v2** : images plus nettes, micro-détails plus stables, meilleure conservation de l'identité et de l'éclairage ; diversité intra-condition plus élevée et respect plus strict des attributs. Les mesures internes indiquent un FID plus bas et un LPIPS légèrement supérieur, traduisant un meilleur compromis fidélité/diversité.

12.2 Contrôle des attributs et désentrelacement

- **v1** : contrôle via prompts CLIP ; sensible à l'ambiguïté textuelle, interactions non désirées entre attributs (p.ex. âge et texture de peau).
- **v2** : vecteur conditionnel explicite et **directions latentes** apprises pour âge, genre et ethnies ; ajustements continus plus *propres* dans W/W^+ , meilleure indépendance des facteurs et moindre *entanglement*.

12.3 Stabilité et robustesse d'entraînement

- **v1** : pertes G/D plus oscillatoires, besoin d'ADA plus fréquent sur sous-groupes rares.
- **v2** : **R1** mieux calibré, discriminant à projection et **EMA** conduisent à une convergence plus stable ; ADA requis à des taux plus faibles ; moins de risque de *mode collapse*.

12.4 Latence, déploiement et coût opérationnel

- **Latence** : génération en une passe pour les deux ; **v2** ajoute un très léger surcoût dû aux embeddings et aux directions, négligeable en usage interactif.
- **Déploiement** : même binaire Gradio ; chargement des checkpoints `sg2_v1_ffhq_cond.pkl` et `sg2_v2_ft_fairface_ema.pkl` ; intégrable dans un conteneur unique.

12.5 Biais et équité

- **v1** : plus sensible aux déséquilibres de distribution et aux prompts biaisés.
- **v2** : bénéfice du *fine-tuning* sur **FairFace** équilibré ; meilleure fidélité attributaire par sous-groupes ; nécessité de *reporting* systématique et d'audits visuels.

12.6 Impact énergétique et écologique

- **Entraînement** : **v2** requiert un temps total plus élevé (pré-entraînement + *fine-tuning*) mais réduit le nombre d'essais nécessaires grâce à une stabilité accrue.
- **Inférence** : coûts similaires entre v1 et v2 ; demi-précision recommandée ; réutilisation des checkpoints conseillée.

12.7 Limites connues et pistes d'amélioration

- **v1** : manque d'axes latents supervisés ; résultats plus variables selon le texte ; améliorer par ajout de directions attributaires et par pondération adaptative du guidage CLIP.
- **v2** : qualité moindre sur classes très rares ou conditions extrêmes ; renforcer par pertes de séparation d'attributs, équilibrage dynamique de lots et régularisations orthogonales dans W^+ .
- **Généralisation** : consolidation du support *non-humain* (animaux) par mélange de corpus et contrôles de sûreté ; évaluation par sous-tâches locales (bouche/yeux) pour l'édition fine.

Références

- [1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Nets. *NeurIPS*.
- [2] Mirza, M., & Osindero, S. (2014). Conditional Generative Adversarial Nets. *arXiv :1411.1784*.
- [3] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *NeurIPS*. (Introduit le FID)
- [4] Karras, T., Laine, S., & Aila, T. (2019). A Style-Based Generator Architecture for Generative Adversarial Networks. *CVPR*. (StyleGAN)
- [5] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and Improving the Image Quality of StyleGAN. *CVPR*. (StyleGAN2)
- [6] Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., & Aila, T. (2020). Training Generative Adversarial Networks with Limited Data. *NeurIPS*. (StyleGAN2-ADA)
- [7] Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral Normalization for Generative Adversarial Networks. *ICLR*. (Stabilisation de StyleGAN2)
- [8] Miyato, T., & Koyama, M. (2018). cGANs with Projection Discriminator. *ICLR*. (Conditionnement avec discriminateur projeté)
- [9] Mescheder, L., Geiger, A., & Nowozin, S. (2018). Which Training Methods for GANs do actually Converge ? *ICML*. (R1 gradient penalty)
- [10] Brock, A., Donahue, J., & Simonyan, K. (2019). Large Scale GAN Training for High Fidelity Natural Image Synthesis. *ICLR*. (Bonnes pratiques grande échelle utiles à SG2)
- [11] Shen, Y., Gu, J., Tang, X., & Zhou, B. (2020). InterFaceGAN : Interpreting the Disentangled Face Representation Learned by GANs. *CVPR*. (Directions latentes âge/genre/etc. pour SG2)
- [12] Häkkinen, E., Hertzmann, A., Lehtinen, J., & Paris, S. (2020). GANSpace : Discovering Interpretable GAN Controls. *NeurIPS*. (Contrôles latents SG2)
- [13] Abdal, R., Qin, Y., & Wonka, P. (2019). Image2StyleGAN : How to Embed Images Into the StyleGAN Latent Space ? *ICCV*. (Inversion pour édition SG2)
- [14] Abdal, R., Qin, Y., & Wonka, P. (2020). Image2StyleGAN++ : How to Edit the Embedded Images ? *TPAMI*. (Édition avancée SG2)
- [15] Tov, O., Alaluf, Y., Patashnik, O., Nitzan, Y., Bermano, A. H., Cohen-Or, D., & Lischinski, D. (2021). Designing an Encoder for StyleGAN Image Manipulation. *SIGGRAPH*. (e4e pour SG2)
- [16] Kynkäanniemi, T., Karras, T., Laine, S., Lehtinen, J., & Aila, T. (2019). Improved Precision and Recall Metric for Assessing Generative Models. *NeurIPS*. (Évaluation complémentaire à FID)
- [17] Karras, T., Aittala, M., Laine, S., Häkkinen, E., Hellsten, J., Lehtinen, J., & Aila, T. (2021). Alias-Free Generative Adversarial Networks. *NeurIPS*. (StyleGAN3, contexte comparatif)
- [18] Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. *NeurIPS*. (Référence diffusion)

Annexes

A Mini-approche StyleGAN2 conditionnel (extrait de code)

Générateur et discriminateur (version minimale)

Listing 1 – SG2 minimal conditionnel : générateur et discriminateur avec discriminateur projeté + R1.

```
import torch, torch.nn as nn, torch.nn.functional as F

AGE_BINS, GENDER_BINS, ETH_BINS = 5, 2, 7
C_DIM = AGE_BINS + GENDER_BINS + ETH_BINS
Z_DIM, W_DIM = 512, 512
IMG_RES, IMG_CH = 256, 3

class MappingNet(nn.Module):
    def __init__(self, in_dim=Z_DIM+C_DIM, out_dim=W_DIM, num_fc=8):
        super().__init__()
        layers=[]
        d=in_dim
        for _ in range(num_fc):
            layers += [nn.Linear(d, out_dim), nn.LeakyReLU(0.2,
                inplace=True)]
            d = out_dim
        self.net = nn.Sequential(*layers)
    def forward(self, z, c):
        x = torch.cat([z, c], dim=1)
        x = x / (x.norm(dim=1, keepdim=True)+1e-8)
        return self.net(x)

class SynthesisMini(nn.Module):
    def __init__(self, w_dim=W_DIM, base=64):
        super().__init__()
        ch = [512,256,128,64,32]
        self.fc = nn.Linear(w_dim, 4*4*ch[0])
        self.blocks = nn.ModuleList([
            nn.Sequential(nn.Upsample(scale_factor=2, mode="nearest"),
            nn.Conv2d(ch[0], ch[1], 3, padding=1), nn.LeakyReLU(0.2, True)),
            nn.Sequential(nn.Upsample(scale_factor=2, mode="nearest"),
            nn.Conv2d(ch[1], ch[2], 3, padding=1), nn.LeakyReLU(0.2, True)),
            nn.Sequential(nn.Upsample(scale_factor=2, mode="nearest"),
            nn.Conv2d(ch[2], ch[3], 3, padding=1), nn.LeakyReLU(0.2, True)),
```

```

        nn.Sequential(nn.Upsample(scale_factor=2, mode="nearest")
                      ,
                      nn.Conv2d(ch[3], ch[4], 3, padding=1), nn.
                      LeakyReLU(0.2, True)),
                ])
        self.to_rgb = nn.Conv2d(ch[4], IMG_CH, 1)
    def forward(self, w):
        x = self.fc(w).view(-1, 512, 4, 4)
        for b in self.blocks: x = b(x)
        return torch.tanh(self.to_rgb(x))

class Generator(nn.Module):
    def __init__(self):
        super().__init__()
        self.mapping = MappingNet()
        self.synth = SynthesisMini()
    def forward(self, z, c):
        w = self.mapping(z, c)
        img = self.synth(w)
        return img

class DiscriminatorProj(nn.Module):
    def __init__(self, c_dim=C_DIM):
        super().__init__()
        ch=[64,128,256,512]
        self.conv = nn.Sequential(
            nn.Conv2d(IMG_CH, ch[0], 3, stride=2, padding=1), nn.
            LeakyReLU(0.2,True),
            nn.Conv2d(ch[0], ch[1], 3, stride=2, padding=1), nn.
            LeakyReLU(0.2,True),
            nn.Conv2d(ch[1], ch[2], 3, stride=2, padding=1), nn.
            LeakyReLU(0.2,True),
            nn.Conv2d(ch[2], ch[3], 3, stride=2, padding=1), nn.
            LeakyReLU(0.2,True),
            nn.AdaptiveAvgPool2d(1)
        )
        self.fc = nn.Linear(ch[-1], 1)
        self.embed = nn.Linear(c_dim, ch[-1], bias=False) # e(y)
    def forward(self, x, c):
        h = self.conv(x).flatten(1) # phi(x)
        out = self.fc(h) # h(x)
        proj = (h * self.embed(c)).sum(1, keepdim=True) # <phi(x), e
        (y)>
        return out + proj

def d_r1_loss(d_out_real, x_real):
    grad = torch.autograd.grad(
        outputs=d_out_real.sum(), inputs=x_real, create_graph=True
    )[0]
    return (grad.flatten(1).pow(2).sum(1)).mean()

```

```

# --- Une itération d'entraînement (non-ADA, non-EMA) ---
G, D = Generator().cuda(), DiscriminatorProj().cuda()
optG = torch.optim.Adam(G.parameters(), lr=2.5e-3, betas=(0,0.99))
optD = torch.optim.Adam(D.parameters(), lr=2.0e-3, betas=(0,0.99))

def one_step(x_real, c_onehot, gamma_r1=10.0):
    B = x_real.size(0)
    z = torch.randn(B, Z_DIM, device=x_real.device)

    # -- D-step
    x_real.requires_grad_(True)
    d_real = D(x_real, c_onehot)
    loss_dr = F.softplus(-d_real).mean()
    r1 = d_r1_loss(d_real, x_real) * (gamma_r1 * 0.5)
    with torch.no_grad():
        x_fake = G(z, c_onehot)
    loss_df = F.softplus(D(x_fake, c_onehot)).mean()
    loss_D = loss_dr + loss_df + r1
    optD.zero_grad(); loss_D.backward(); optD.step()

    # -- G-step
    z = torch.randn(B, Z_DIM, device=x_real.device)
    x_fake = G(z, c_onehot)
    loss_G = F.softplus(-D(x_fake, c_onehot)).mean()
    optG.zero_grad(); loss_G.backward(); optG.step()
    return loss_D.item(), loss_G.item()

```

Encodage des attributs (exemple FairFace)

Listing 2 – Encodage one-hot simple pour âge, genre et ethnique.

```

def encode_attrs(age_bin, gender, eth_idx):
    # age_bin in {0..4}, gender in {0=Male,1=Female}, eth_idx in
    # {0..6}
    age = F.one_hot(torch.tensor([age_bin]), num_classes=5).float()
    gen = F.one_hot(torch.tensor([gender]), num_classes=2).float()
    eth = F.one_hot(torch.tensor([eth_idx]), num_classes=7).float()
    return torch.cat([age, gen, eth], dim=1) # shape (1,14)

```