



## RAPPORT DE PROJET

JUIN 2025

**4IM06**

# **SinGAN : Learning a Generative Model from a Single Natural Image**

MARTIN DE BATS  
CLÉMENT GILLI  
ROMAIN PLANCHON  
QUENTIN RÉVILLON

ENCADRANTS : YANN GOUSSEAU, ARTHUR LECLAIRE

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Méthode</b>	<b>2</b>
2.1	Architecture . . . . .	2
2.2	Entraînement . . . . .	3
<b>3</b>	<b>Analyse des résultats</b>	<b>5</b>
3.1	Visualisation qualitative . . . . .	5
3.2	Distances sur les distributions de features . . . . .	6
3.3	Visualisation des distributions : t-SNE . . . . .	7
3.4	Limites observées . . . . .	8
3.5	Évaluation perceptuelle avec LPIPS . . . . .	8
3.6	Cas des textures . . . . .	9
3.7	Conclusion . . . . .	10
<b>4</b>	<b>Implémentation</b>	<b>11</b>
<b>5</b>	<b>Peut-on se passer d'un discriminateur ?</b>	<b>13</b>
5.1	Pourquoi remplacer le couple générateur-discriminateur habituel ? . . . . .	13
5.2	Pourquoi des features Inception plutôt que des patchs pixels ? . . . . .	13
5.3	Espace de comparaison retenu . . . . .	13
5.4	Fonction de perte . . . . .	13
5.5	Convergence empirique . . . . .	14
5.6	Résultats et discussion . . . . .	15
5.7	Cas des textures . . . . .	16
<b>6</b>	<b>Comparaison des résultats</b>	<b>17</b>
6.1	Résultats qualitatifs . . . . .	17
6.2	Résultats quantitatifs . . . . .	17
6.3	Analyse . . . . .	18
6.4	Cas des textures . . . . .	18
<b>7</b>	<b>Conclusion</b>	<b>19</b>
	<b>Bibliographie</b>	<b>20</b>
	<b>Annexe</b>	<b>21</b>

# 1 Introduction

La génération d’images réalistes est un domaine de recherche central dans le domaine de l’apprentissage profond, avec de nombreuses applications dans la vision par ordinateur, la synthèse d’images et l’augmentation des données. Les modèles génératifs adversariaux (GAN) permettent de générer des échantillons de haute qualité à partir d’une distribution latente apprise. Cependant, la plupart des GAN nécessitent de grands ensembles de données pour capturer une distribution suffisamment riche.

Dans ce contexte, l’article ”SinGAN : Learning a Generative Model from a Single Natural Image” [1] propose une approche radicalement différente : l’apprentissage d’un modèle génératif réaliste à partir d’une seule image. SinGAN est basé sur une hiérarchie de GAN entraînés à différentes échelles, chaque niveau apprenant les structures locales spécifiques à son échelle.

L’objectif de ce projet est multiple. Tout d’abord, nous avons réimplémenté SinGAN en suivant les principes de l’article original. Nous nous sommes ensuite concentrés sur la simplification du modèle et de l’entraînement en expérimentant de nouvelles fonctions de perte. Enfin, nous avons proposé une évaluation quantitative de la qualité des images générées à l’aide de différentes mesures statistiques (SIFID, Wasserstein distance, SWD, LPIPS).

## 2 Méthode

Le problème de générer des images à partir d’une seule image pose un défi important. Si le réseau est trop puissant, il apprend rapidement à copier exactement l’image d’origine sans générer de variations. Au contraire, si le réseau est trop simple, il ne parvient pas à reproduire les structures globales et les motifs visuels présents dans l’image.

SinGAN propose une solution à ce compromis. L’idée principale est de ne pas apprendre l’image entière d’un coup, mais de l’apprendre progressivement à travers une pyramide d’échelles. À chaque échelle, un petit GAN est entraîné pour apprendre uniquement les motifs visuels propres à cette résolution.

Les réseaux apprennent donc à générer des patchs réalistes de l’image à leur échelle, sans mémoriser l’image complète. Cette approche multi-scale permet d’éviter le sur-apprentissage et d’apprendre des motifs visuels cohérents à plusieurs niveaux. Elle permet ensuite de générer des images réalistes et variées qui respectent le style et les structures de l’image d’origine.

### 2.1 Architecture

Le modèle complet est constitué d’une série de générateurs  $\{G_0, \dots, G_N\}$  et de discriminateurs  $\{D_0, \dots, D_N\}$ . Chaque paire  $(G_n, D_n)$  forme un GAN indépendant, entraîné à une certaine échelle de l’image d’origine  $x$ , réduite d’un facteur  $r^{N-n}$ , où  $r > 1$  est un paramètre contrôlant la progression entre les différentes résolutions.

Comme dans un GAN classique, le générateur  $G_n$  tente de produire des images capables de tromper le discriminateur  $D_n$ , lequel apprend à distinguer les vrais patchs de ceux générés. Toutefois, chaque générateur ne part pas de zéro : il prend en entrée une image générée à l’échelle précédente, interpolée à la taille courante, ainsi qu’un bruit aléatoire  $z_n$ . La sortie du générateur  $G_n$  à l’échelle  $n$  est alors définie par :

$$\tilde{x}_n = \tilde{x}_{n-1}^{\uparrow r} + G_n(z_n + \tilde{x}_{n-1}^{\uparrow r})$$

où  $\tilde{x}_{n-1}^{\uparrow r}$  est l’image générée à l’échelle précédente, redimensionnée par un facteur  $r$ .

Cette architecture permet à chaque niveau de raffiner les détails ajoutés par les échelles précédentes, tout en injectant de la nouveauté via le bruit  $z_n$ . À la plus petite échelle (niveau 0), aucune image précédente n’est disponible, donc le générateur  $G_0$  agit comme un GAN classique, ne prenant en entrée que le bruit  $z_0$ .

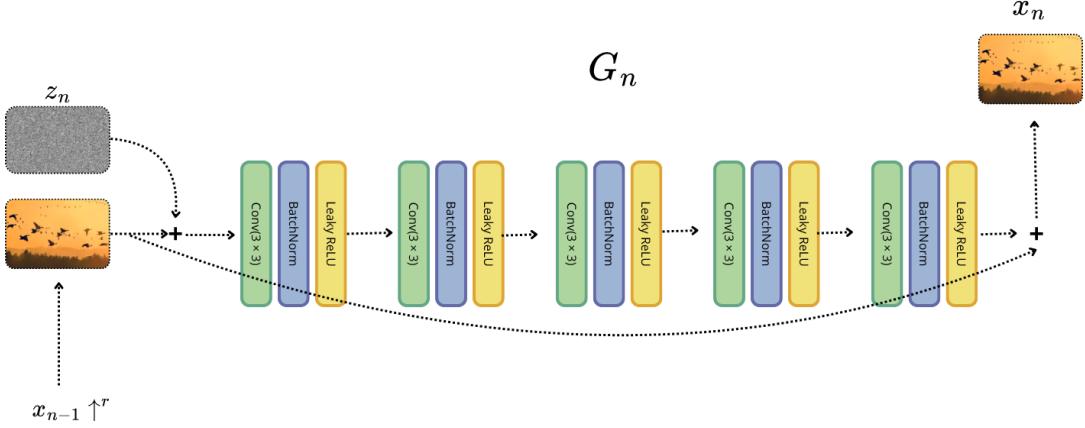


FIGURE 1 – Architecture multi-échelle de SinGAN. Chaque générateur affine les détails produits par le précédent.

L’architecture de chaque générateur est identique : elle comporte 5 blocs convolutionnels, chacun constitué d’une convolution  $3 \times 3$ , suivie d’une normalisation de lot (batch normalization) et d’une activation Leaky ReLU (Figure 1). Pour éviter que les générateurs à petites échelles n’apprennent à reproduire l’image en entier, la capacité du réseau est modulée en fonction de l’échelle. Le nombre de canaux est défini par :

$$k_n = 32 \times 2^{\lfloor \frac{N-n}{4} \rfloor}$$

Cela permet de limiter la complexité des premiers générateurs et d’augmenter progressivement la capacité du modèle à mesure que la résolution augmente.

Les discriminateurs  $\{D_0, \dots, D_N\}$  partagent une architecture similaire à celle des générateurs, composée de blocs convolutionnels suivis de normalisation et d’activation. À cela s’ajoute, en sortie, un bloc de *Global Average Pooling* suivi d’une activation sigmoïde. Cette structure permet de transformer la prédiction locale, basée sur des patchs effectifs de taille  $11 \times 11$ , en une prédiction globale, fournissant ainsi une estimation binaire de l’image à l’échelle considérée.

## 2.2 Entrainement

L’entraînement du modèle s’effectue de manière hiérarchique, en partant de l’échelle la plus basse vers la plus haute (Figure 2). À chaque niveau  $n$ , le couple  $(G_n, D_n)$  est entraîné indépendamment, tandis que les générateurs précédents  $\{G_0, \dots, G_{n-1}\}$  sont fixés. Ces derniers servent à produire, par upsampling et génération successive, une image intermédiaire  $\tilde{x}_{n-1}$  utilisée comme entrée pour l’échelle courante.

Chaque générateur est entraîné selon une perte de type WGAN-GP, qui assure la stabilité de l’apprentissage adversarial, à laquelle s’ajoute une contrainte de reconstruction. Cette perte de reconstruction veille à ce que l’image originale  $x$  appartienne bien aux images que le modèle peut générer à partir du bruit initial. Le problème d’optimisation considéré à chaque échelle est alors le suivant :

$$\min_{G_n} \max_{D_n} \mathcal{L}_{\text{WGAN-GP}}(G_n, D_n) + \alpha \mathcal{L}_{\text{rec}}(G_n)$$

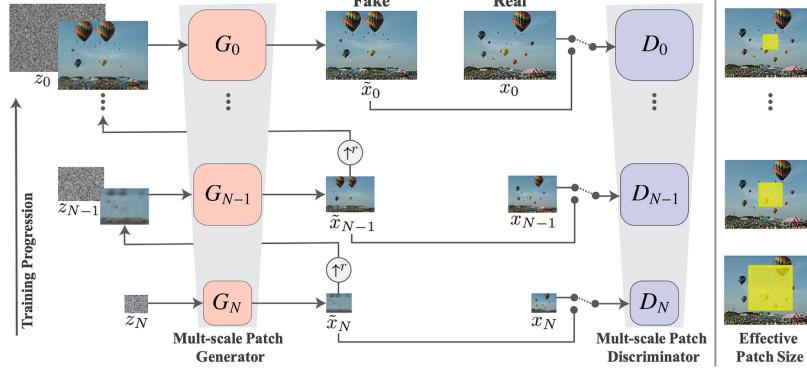


FIGURE 2 – SinGAN training pipeline

La loss de reconstruction repose sur l'idée de fixer un bruit  $z^{\text{rec}}$  constant à travers l'ensemble des niveaux de génération, de manière à guider progressivement le réseau vers la reconstitution fidèle de l'image cible.

Par souci de simplicité, le vecteur de bruit est défini comme  $z^{\text{rec}} = \{z^*, 0, \dots, 0\}$ , où  $z^*$  est un bruit aléatoire initial à la plus petite échelle. À chaque niveau  $n$ , on génère une image intermédiaire  $x_n^{\text{rec}}$  en injectant le bruit et les reconstructions précédentes :

$$x_n^{\text{rec}} = G_n(z_n^{\text{rec}}, x_{n-1}^{\text{rec}} \uparrow^r),$$

La loss de reconstruction s'écrit alors comme une erreur quadratique moyenne entre l'image d'entrée  $x_n$  à l'échelle courante et sa reconstruction  $x_n^{\text{rec}}$  :

$$\mathcal{L}_{\text{rec}} = \|x_n - x_n^{\text{rec}}\|_2^2.$$

Enfin, la loss de reconstruction fournit également une estimation de la quantité de bruit nécessaire à injecter à chaque niveau de génération pour produire de nouveaux détails tout en restant cohérent avec l'image cible. L'idée est d'ajuster dynamiquement l'échelle du bruit  $z_n$  en fonction de l'erreur de reconstruction entre l'image originale à l'échelle  $n$  et la reconstruction obtenue à partir de l'échelle précédente. Plus précisément, on définit l'écart type  $\sigma_n$  du bruit injecté à l'échelle  $n$  comme étant proportionnel à l'erreur quadratique moyenne entre  $x_n$  et  $x_{n-1}^{\text{rec}}$  upsamplée :

$$\sigma_n = \lambda_{\text{amp noise}} * \sqrt{\text{MSE}(x_n, x_{n-1}^{\text{rec}} \uparrow^r)}.$$

$$z_n \sim \mathcal{N}(0, \sigma_n^2).$$

Cette stratégie permet au réseau d'introduire une quantité de bruit proportionnelle à la quantité d'information manquante à chaque échelle.

### 3 Analyse des résultats

Dans le cadre de ce projet, il était important de disposer d'un critère quantitatif permettant d'évaluer la qualité des images générées, et ce de manière plus objective qu'un simple jugement visuel ("l'image est jolie"). L'objectif était donc de trouver une métrique capable de quantifier à quel point une image générée est proche, qualitativement, de l'image de référence.

#### 3.1 Visualisation qualitative

Avant d'aborder les métriques quantitatives, nous montrons ci-dessous quelques exemples d'images générées par SinGAN, à partir d'une seule image réelle (Figure 3). Ces visualisations permettent d'illustrer visuellement la qualité (ou les défauts) des résultats produits.



Image originale

Images générées

FIGURE 3 – Exemples de générations SinGAN. Chaque ligne correspond à une image d'entraînement (première colonne), suivie de plusieurs échantillons générés aléatoirement à partir de cette image.

Nous observons que les images générées d'oiseaux et d'herbe sont plutôt visuellement acceptables, tandis que les images aériennes de Londres générées ainsi que les images de zèbre générées sont complètement absurdes. Nous observons une première limite de SinGAN : la génération marche sur des cas bien spécifiques (du moins lorsqu'on génère à partir de zéro).

### 3.2 Distances sur les distributions de features

Une première piste explorée fut le FID (Fréchet Inception Distance), couramment utilisé pour évaluer la qualité des images générées. Étant donné que SinGAN s'appuie sur une seule image d'entrée, nous avons utilisé la variante SIFID, qui applique le FID à des patchs locaux extraits d'une image.

L'idée est de passer l'image originale ainsi que les images générées dans un réseau pré-entraîné, en l'occurrence Inception ici, puis de récupérer des features profondes pour les patchs de l'image. Dans le cas d'Inception, on obtient 64 patchs avec 2048 features chacun, on peut donc représenter cela comme un nuage de 64 points de dimension 2048. Il suffit ensuite d'inférer une gaussienne sur ces points pour chaque image passée en entrée, puis de calculer la distance entre l'image originale et les images générées (une à une) à partir de la forme close :

$$d_F(\mathcal{N}(\mu, \Sigma), \mathcal{N}(\mu', \Sigma'))^2 = \|\mu - \mu'\|_2^2 + \text{tr} \left( \Sigma + \Sigma' - 2(\Sigma \Sigma')^{\frac{1}{2}} \right)$$

Cependant, cette métrique repose sur une hypothèse forte : la modélisation des nuages de points (features) par une distribution gaussienne. Cela rend la distance calculable avec une forme close, mais limite sa précision. Pour une estimation plus précise, nous avons envisagé la distance de Wasserstein entre nuages de points non gaussiens :

Étant donnés deux ensembles de features  $\{x_i\}_{i=1}^n$  et  $\{y_j\}_{j=1}^n$  dans  $\mathbb{R}^d$ , la distance de Wasserstein de premier ordre (aussi appelée Earth Mover's Distance) est définie par :

$$W_1(\mu, \nu) = \min_{\gamma \in \Pi(\mu, \nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\| d\gamma(x, y) \quad (1)$$

où  $\mu = \frac{1}{n} \sum_{i=1}^n \delta_{x_i}$  et  $\nu = \frac{1}{n} \sum_{j=1}^n \delta_{y_j}$  sont les mesures empiriques associées aux nuages de points, et  $\Pi(\mu, \nu)$  est l'ensemble des plans de transport (couplages) entre  $\mu$  et  $\nu$ .

Dans la pratique, ce problème est résolu via un algorithme d'assignement optimal (par exemple, l'algorithme de l'hongrois).

Pour réduire le coût de calcul en grande dimension, nous avons testé la Sliced Wasserstein Distance, une approximation par projections 1D aléatoires :

La Sliced Wasserstein distance consiste à projeter les données sur des droites aléatoires unitaires  $\theta \in \mathbb{S}^{d-1}$ , et à calculer la Wasserstein distance dans  $\mathbb{R}$  :

$$SW_1(\mu, \nu) = \int_{\mathbb{S}^{d-1}} W_1(\mathcal{R}_\theta \mu, \mathcal{R}_\theta \nu) d\theta \quad (2)$$

où  $\mathcal{R}_\theta \mu$  est la mesure projetée sur la direction  $\theta$  :

$$\mathcal{R}_\theta \mu = \{\langle x_i, \theta \rangle\}_{i=1}^n$$

En pratique, cette intégrale est approximée par une moyenne sur  $K$  directions aléatoires :

$$SW_1^K(\mu, \nu) = \frac{1}{K} \sum_{k=1}^K W_1(\mathcal{R}_{\theta_k} \mu, \mathcal{R}_{\theta_k} \nu) \quad (3)$$

Toutefois, les mesures empiriques sur notre cas (64 points en dimension 2048) montrent que la version exacte est non seulement plus précise, mais aussi plus rapide (Figure 4). Ainsi, nous allons donc utiliser directement la distance de Wasserstein.

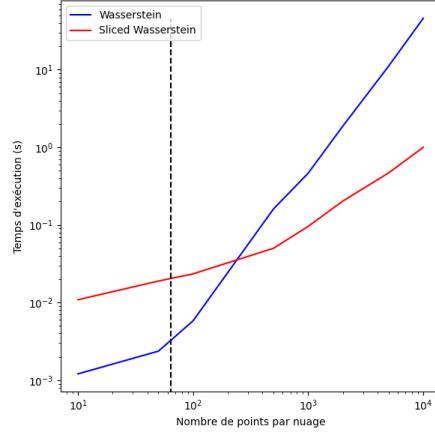


FIGURE 4 – Temps d’exécution (log-échelle) en fonction du nombre de points de dimension 2048 pour Wasserstein vs Sliced Wasserstein. En pointillé le nombre de points qui nous intéresse : 64.

### 3.3 Visualisation des distributions : t-SNE

Afin de visualiser la structure des features extraits par Inception, nous avons projeté les nuages de points (dimension 2048) en 2D via t-SNE (Figure 5). Cette représentation révèle que, dans le cas des images de mauvaise qualité (ex : zèbre), les features générés sont parfois très proches de ceux de l’image réelle, malgré une forte dissemblance visuelle, ce qui est assez problématique pour nous.

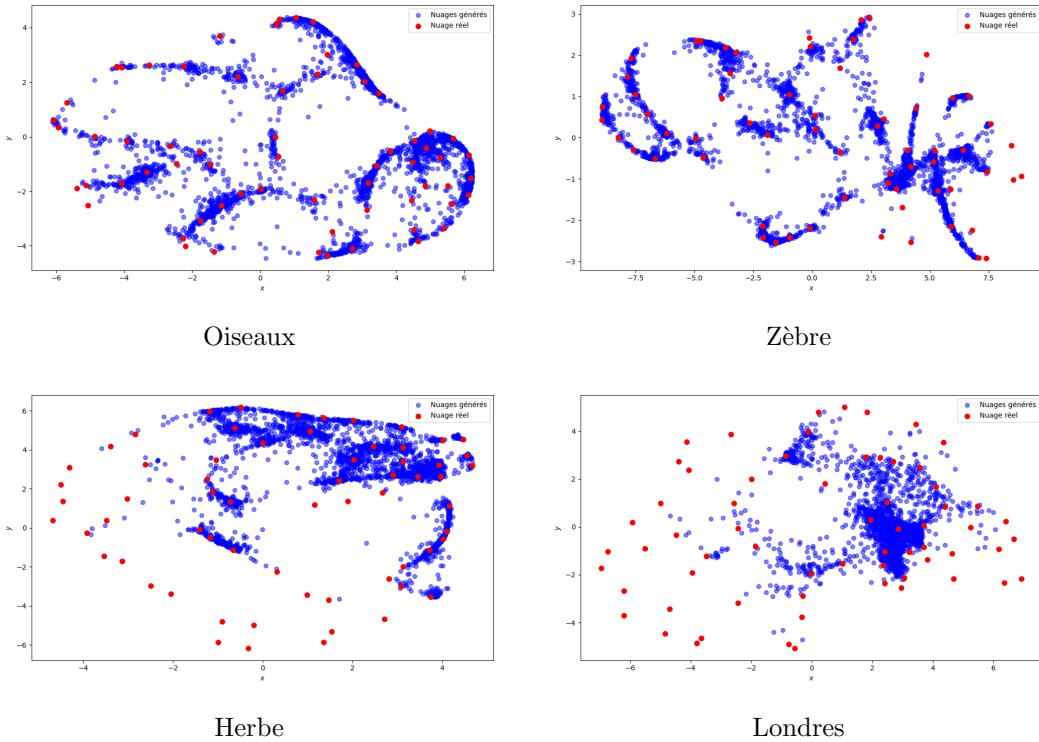


FIGURE 5 – Visualisation t-SNE des features extraits (réels vs générés) pour différentes images de départ. Les points rouges représentent les features de l’image d’origine, les bleus ceux des échantillons générés par SinGAN. À noter que dans chaque figure il y a 50 nuages de points correspondant à 50 images générées différentes, l’idée étant d’échantillonner la distribution apprise par le réseau.

### 3.4 Limites observées

En calculant la distance de Wasserstein entre les features extraites pour 50 images générées et l'image réelle (Table 1), nous avons observé des résultats qui appuient ce que nous avons remarqué avec les représentations t-SNE. Dans le cas du zèbre, les images générées étaient très éloignées visuellement de l'image originale, mais obtenaient parfois de meilleures distances que d'autres cas plus réussis comme les oiseaux.

Image	Moyenne	Écart-type	Min	Max
Herbe	26.469	1.038	23.940	28.574
Oiseaux	26.051	2.445	22.971	33.468
Londres	33.910	1.150	30.969	37.482
Zèbre	13.681	0.957	12.106	15.801

TABLE 1 – Statistiques des distances de Wasserstein entre les nuages de points (features) des images réelles et générées.

Cela indique que ces métriques mesurent essentiellement la proximité locale des features, mais ne sont pas sensibles à la cohérence perceptuelle globale de l'image. C'est notamment problématique dans des cas où l'image générée respecte certaines textures locales mais présente une structure totalement absurde. C'est sûrement pour cela que le zèbre obtient des distances faibles avec l'image d'origine : on a deux textures globales qui sont les rayures de la peau ainsi que l'herbe en arrière-plan.

### 3.5 Évaluation perceptuelle avec LPIPS

Pour pallier cette limite et donc évaluer la qualité perceptuelle des images générées, nous avons utilisé la métrique *LPIPS* (Learned Perceptual Image Patch Similarity), introduite dans [2]. Cette métrique vise à mesurer la similarité entre deux images en se basant encore sur des représentations internes extraites par un réseau convolutif pré-entraîné (tel que VGG ou AlexNet). Concrètement, chaque image est projetée dans l'espace des features de différentes couches du réseau, puis les distances (typiquement euclidiennes) entre ces représentations sont calculées couche par couche. Ces distances sont ensuite pondérées par des coefficients appris, afin de refléter au mieux la perception humaine. Une valeur LPIPS faible indique une forte similarité perceptuelle. Contrairement à la distance de Wasserstein, qui mesure une divergence entre distributions statistiques, LPIPS fournit une mesure directe de la cohérence visuelle et du réalisme perçu entre deux images. Elle s'avère donc complémentaire et souvent plus pertinente pour juger de la qualité générative à l'œil humain.

Les résultats obtenus sont alors bien plus cohérents. L'histogramme (Figure 6) ci-dessous montre une séparation claire entre les distributions des distances LPIPS pour des images de bonne qualité (comme celles des oiseaux) et celles de mauvaise qualité (comme celles du zèbre). On fournit également un tableau récapitulatif des différentes statistiques (Table 7).

Image	Moyenne	Écart-type	Min	Max
Herbe	0.323	0.013	0.292	0.353
Oiseaux	0.373	0.021	0.315	0.418
Londres	0.640	0.012	0.607	0.664
Zèbre	0.472	0.033	0.378	0.541

TABLE 2 – Statistiques des distances LPIPS entre les images réelles et générées. Une valeur plus faible indique une plus grande similarité perceptuelle.

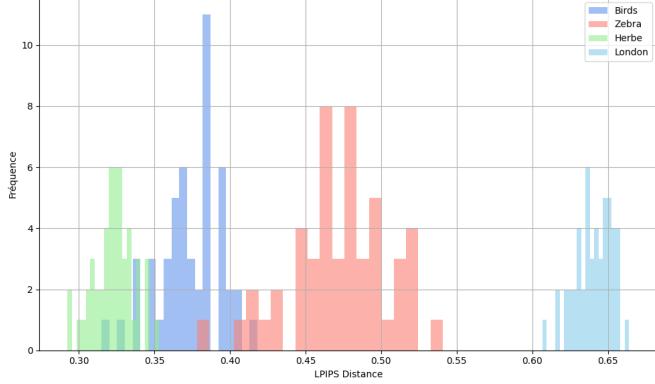


FIGURE 6 – Distribution des distances LPIPS (VGG) entre l'image réelle et les images générées.

### 3.6 Cas des textures

Nous nous sommes également intéressés au cas des textures, car il permet d'observer la faculté de SinGAN à générer. Nous avons pu identifier 3 cas :

- la texture est un motif répété
- la texture est gaussienne
- la texture est non-gaussienne

Pour mieux observer la capacité de SinGAN à synthétiser des textures, nous avons doublé la taille de l'image générée (Figure 7).

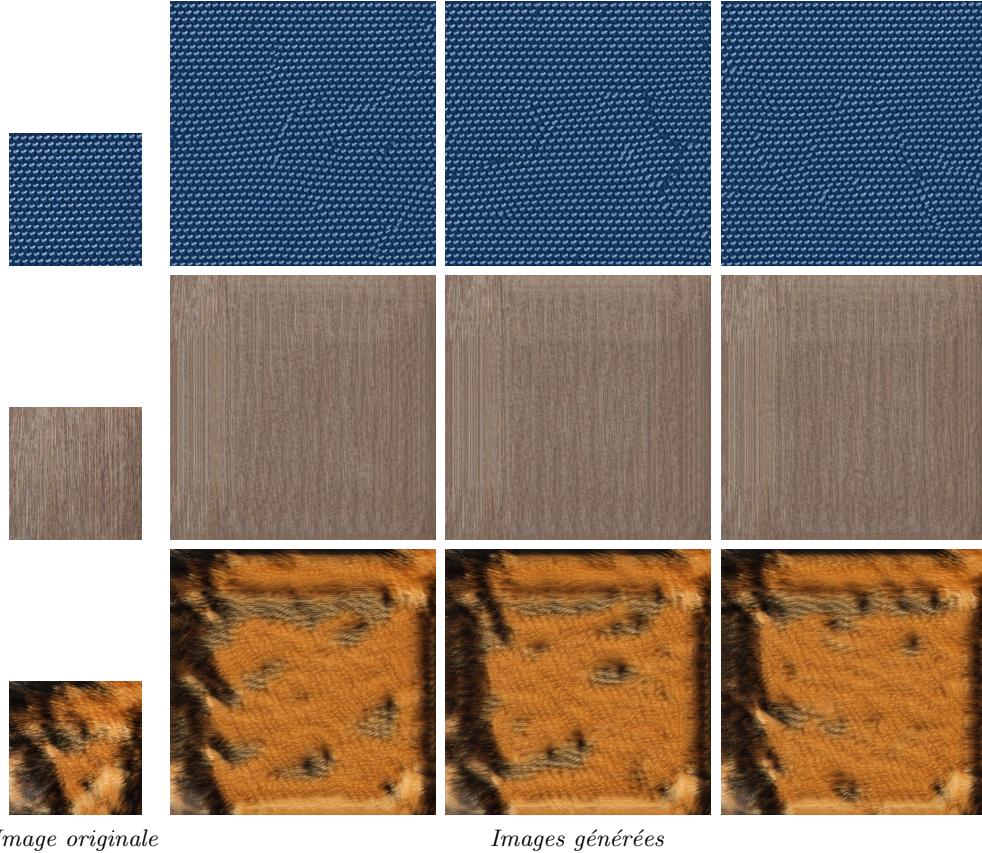


FIGURE 7 – Exemples de générations de texture. Première ligne : motif répété. Deuxième ligne : gaussien. Troisième ligne : non-gaussien

Pour chaque cas, nous avons mesuré la similarité entre les images générées et l'image de base à l'aide des métriques LPIPS (perceptuelle, Table 3) et Wasserstein (statistique, Table 4).

Dans le cas du motif répétitif, bien que la génération reste globalement correcte, elle souffre de défauts de régularité avec des artefacts visibles. Cette impression est reflétée dans les scores : une distance LPIPS moyenne de 0.493 et une distance de Wasserstein de 19.23. Cela montre que SinGAN parvient à capturer une partie de la structure mais peine à restituer un motif parfaitement régulier.

Pour la texture gaussienne, les résultats sont plus ambigus : visuellement, le modèle semble davantage étirer la texture de base que la générer de manière autonome. La métrique LPIPS (0.361) suggère une certaine proximité visuelle avec l'image d'origine, mais la distance de Wasserstein est plus élevée (24.40), indiquant un écart notable au niveau de la distribution des caractéristiques extraites.

Enfin, le cas des textures non-gaussiennes révèle les limites claires du modèle. Visuellement, les images générées apparaissent incohérentes et ne respectent pas la structure de la texture de départ. Cela se confirme par des scores dégradés : une distance LPIPS élevée (0.618) et une distance de Wasserstein atteignant 30.40. Ces valeurs traduisent l'incapacité du modèle à généraliser sur des textures complexes et à forte variabilité statistique.

Ce bilan montre que, bien que SinGAN puisse produire des résultats intéressants sur des textures simples, ses performances se dégradent rapidement dès que la structure devient moins régulière ou statistiquement complexe.

Texture	Moyenne	Écart-type	Min	Max
Motif	0.493	0.004	0.485	0.501
Gaussien	0.361	0.007	0.347	0.374
Non-gaussien	0.618	0.005	0.609	0.629

TABLE 3 – Statistiques LPIPS selon différents types de textures. Une valeur plus faible indique une plus grande similarité perceptuelle entre l'image générée et l'image réelle.

Texture	Moyenne	Écart-type	Min	Max
Motif	19.228	0.280	18.647	19.821
Gaussienne	24.401	0.818	22.634	25.680
Non-gaussienne	30.403	0.788	29.233	32.225

TABLE 4 – Statistiques des distances de Wasserstein entre les features des images générées et de référence pour différents types de textures.

### 3.7 Conclusion

Les distances sur les distributions de features (SIFID, Wasserstein) permettent de mesurer la proximité locale à l'image de référence, mais manquent de sensibilité à la qualité perceptuelle globale. À l'inverse, la métrique LPIPS s'avère particulièrement efficace pour capturer les écarts visuels perçus par un humain. Elle constitue donc un outil pertinent et complémentaire dans notre contexte pour évaluer la qualité des images générées par SinGAN.

## 4 Implémentation

Dans le cadre de ce projet, nous avons réimplémenté *ex nihilo* l'architecture et le protocole d'entraînement décrits dans SinGAN [1], sans recourir au code officiel. Les hyperparamètres absents du papier ont toutefois été repris depuis le répertoire d'origine afin de rester comparables.

**Pourquoi réimplémenter de cette manière ?** L'objectif n'était pas de répliquer bit-à-bit le dépôt officiel, mais en procédant de cette façon de comprendre en profondeur l'architecture et le rôle de chaque hyperparamètre.

Pour vérifier la fidélité de notre reproduction, nous avons généré des échantillons à partir des mêmes images d'entrée et les avons comparés visuellement à ceux de l'implémentation auteur. La Figure 8 juxtapose, pour quatre exemples, l'image réelle et le résultat issu de notre code. Les textures, les structures globales et la variété des motifs sont quasi identiques, ce qui confirme la robustesse de notre réimplémentation. Concernant les textures la figure ?? montre des résultats similaires au cas des textures analysées dans la partie 3.6

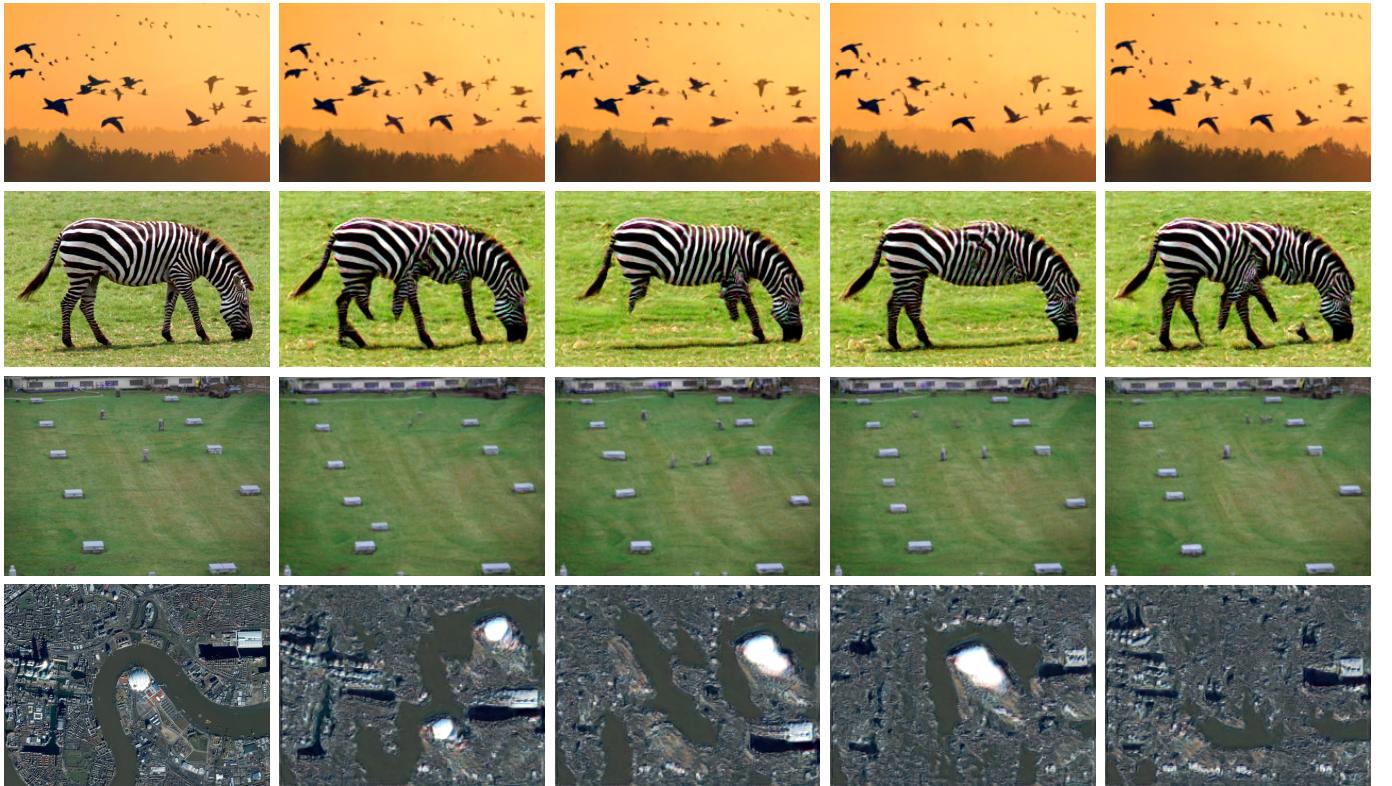


FIGURE 8 – Exemples de générations avec notre implémentation.

On constate que les résultats obtenus à partir de notre implémentation sont très proches de ceux du papier. La qualité visuelle est préservée, les textures sont cohérentes avec l'image source, et les motifs générés sont variés tout en respectant les contraintes locales. Ces observations confirment la robustesse de notre implémentation, qui parvient à reproduire le comportement attendu du modèle SinGAN.

c

### Quelques leçons tirées de la ré-implémentation

1. **Rôle important de la perte de reconstruction.** Nous pensions au départ que la Loss de reconstruction s'agissait d'une régularisation mineure alors que sans elle, la conver-

gence est tout simplement absente.

2. **Régénération du volume intermédiaire  $\tilde{x}$ .** Le papier ne précise pas que l'image intermédiaire doit être *re-générée à chaque itération* en remontant toute la pyramide ; si l'on réutilise un  $\tilde{x}$  figé, chaque générateur n'apprend qu'à partir d'un unique exemple et la généralisation disparaît. C'est assez intuitif mais assez étonnant que non mentionné dans le papier.
3. **Gestion des bords.** Pour la simplicité, nous avons appliqué un *padding* qui préserve la taille des images à toutes les convolutions, au lieu de respecter les conditions aux bords proposées dans l'article. Cette variante tend à reproduire davantage les pixels initiaux près des bordures, ce qui réduit légèrement la diversité locale.

## 5 Peut-on se passer d'un discriminateur ?

### 5.1 Pourquoi remplacer le couple générateur-discriminateur habituel ?

Dans SinGAN, chaque échelle possède son propre discriminateur, formant une boucle adversariale locale avec le générateur. Même légers, ces discriminateurs impliquent des contraintes supplémentaires : double optimisation antagoniste, *gradient penalty*, et une consommation mémoire non négligeable. Nous proposons ici de supprimer entièrement cette branche, et de guider le générateur à l'aide d'une distance de transport optimal, la *Sliced Wasserstein Distance (SWD)*, évaluée dans un espace de représentation perceptuelle dérivé d'Inception-v3, comme introduite précédemment 2. Le schéma multi-échelle reste identique à celui de SinGAN.

### 5.2 Pourquoi des features Inception plutôt que des patchs pixels ?

Comparer deux images au niveau pixel, ou même au niveau de petits patchs RGB, se révèle trop fragile. Une translation de quelques pixels ou une variation globale de luminosité suffit à générer une erreur significative. De plus, la pénalisation se fait indépendamment sur chaque canal couleur, ignorant la structure perceptuelle globale de l'image. Ce type de mesure ne capture pas la hiérarchie des motifs visuels, des textures locales jusqu'au contenu sémantique global.

À l'inverse, les activations internes d'un réseau comme Inception-v3 offrent une représentation plus robuste. Elles présentent une invariance modérée aux petites transformations géométriques et photométriques, et codent l'information à plusieurs niveaux d'abstraction. Plus on progresse dans le réseau, plus la représentation devient sémantique.

Il ne s'agit pas ici de changer drastiquement l'architecture de SinGAN. Le discriminateur jouait déjà le rôle implicite d'un filtre perceptuel, en apprenant une métrique adaptée via ses couches de convolution, normalisation et activation. Nous remplaçons simplement ce filtre appris par un filtre fixe, celui d'Inception-v3 (pré-entraîné sur ImageNet), et mesurons directement la distance dans cet espace intermédiaire.

### 5.3 Espace de comparaison retenu

Pour capturer l'information à plusieurs échelles de perception, nous sélectionnons trois couches internes d'Inception :

- **Mixed\_5d** ( $35 \times 35$ , 288 canaux) pour les détails fins
- **Mixed\_6e** ( $17 \times 17$ , 768 canaux) pour la structure intermédiaire
- **Mixed\_7c** ( $8 \times 8$ , 2048 canaux) pour le contenu global.

Pour un batch  $B$ , chaque tenseur de forme  $(B, C, H, W)$  est réarrangé en nuage de points  $(BHW, C)$ ; par exemple,  $(B, 2048, 8, 8)$  devient  $(B \times 64, 2048)$  pour la couche **Mixed\_7c**. La distance SWD est ensuite évaluée entre les distributions de features ainsi obtenues.

### 5.4 Fonction de perte

L'apprentissage repose donc sur l'alignement entre les distributions de features extraites d'Inception pour les images générées et pour l'image d'entrée. À chaque itération, pour chaque couche retenue (**Mixed\_5d**, **Mixed\_6e**, **Mixed\_7c**), on calcule la Sliced Wasserstein Distance (SWD) entre les nuages de vecteurs correspondants. Ces distances sont ensuite combinées linéairement en une loss totale :

$$\mathcal{L}_{\text{tot}} = \sum_{k \in \{5d, 6e, 7c\}} \lambda_k \text{SWD}(F_{\text{fake}}^{(k)}, F_{\text{real}}^{(k)}) + \alpha \mathcal{L}_{\text{rec}}(G_n).$$

Les coefficients  $\lambda_k$  sont fixés manuellement selon une rampe déterministe qui évolue au fil des itérations d'une même échelle : on privilégie d'abord la structure globale en accordant plus

de poids à la couche profonde `Mixed_7c`, puis on accorde progressivement plus d'importance aux couches intermédiaires et superficielles pour affiner les détails. Typiquement, la pondération évolue de  $[0.4, 0.7, 1.0]$  à  $[1.0, 0.7, 0.4]$ . Ces valeurs ne sont pas apprises, mais elles sont suffisantes pour fournir un guidage cohérent et progressif à travers les niveaux de détail.

Un terme de reconstruction  $\mathcal{L}_{\text{rec}}$  est également conservé, comme dans l'implémentation originale de SinGAN, afin d'assurer que le bruit injecté à l'échelle la plus fine permette de retrouver précisément l'image source.

## 5.5 Convergence empirique

L'évolution des pertes SWD au cours de l'apprentissage (Figure 11) montre une décroissance progressive puis une stabilisation pour les trois couches considérées. Ce comportement indique un rapprochement entre les distributions de features réelles et synthétiques, bien que l'on observe souvent un plateau précoce : passé un certain point, le générateur n'améliore plus significativement la concordance statistique, malgré des itérations supplémentaires. Cela témoigne à la fois d'une certaine efficacité initiale du guidage par SWD, mais aussi d'une limite de cette méthode qui ne pousse pas à une reproduction fine et exacte des structures.

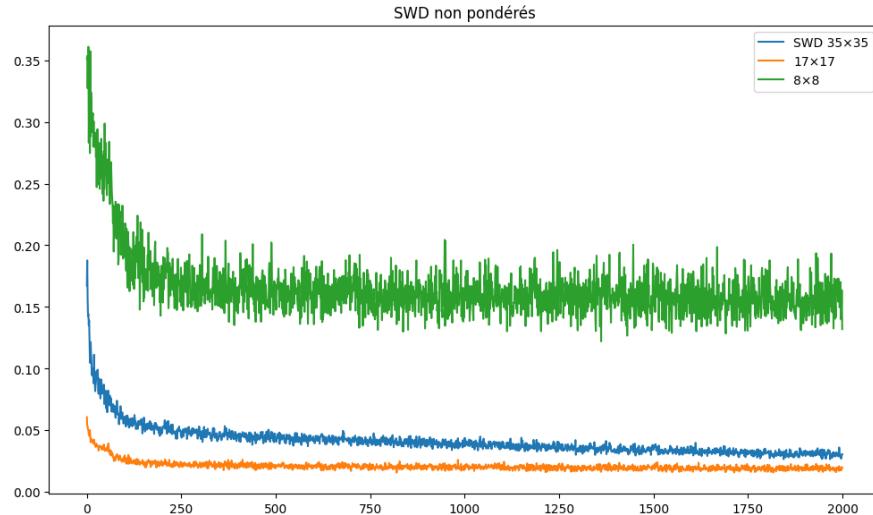


FIGURE 11 – Évolution des trois SWD ( $8 \times 8$ ,  $17 \times 17$ ,  $35 \times 35$ ) lors de l'apprentissage sur l'image *oiseau*.

## 5.6 Résultats et discussion



*Image originale*

*Images générées*

FIGURE 12 – Exemples de générations avec une loss SWD. Chaque ligne correspond à une image d’entraînement (première colonne), suivie de plusieurs échantillons générés aléatoirement à partir de cette image.

Le remplacement du discriminateur par une mesure SWD dans l'espace de features présente plusieurs avantages :

**Rapidité et consommation mémoire :** L'apprentissage devient sensiblement plus rapide : à nombre d'échelles constant nous observons  $\approx 40\%$  de temps gagné, et la mémoire GPU diminue puisque l'on supprime toute phase adversariale (rétro-propagation du critique, contrainte Lipschitz, etc.).

**Réduction du nombre d'échelles :** Un second bénéfice, plus marquant encore, est la réduction du nombre d'échelles nécessaires. Le papier SinGAN[1] recommandait un facteur de zoom  $r = \frac{4}{3}$  avec des tailles minimales et maximales de 25 px et 250 px, ce qui conduit le plus souvent à six à huit niveaux. Dans notre configuration "SWD multi-échelle", **trois ou quatre niveaux suffisent** pour atteindre la même qualité visuelle que l'on obtiendrait en conservant davantage de résolutions ; l'entraînement complet d'une image peut alors passer sous les cinq-sept minutes sans réelle optimisation supplémentaire du code.

**Pistes d'accélération supplémentaires :** Sur ce point nous avons pensé à quelques stratégies pour encore améliorer la vitesse d'entraînement sans perte de qualité :

1. la courbe SWD se stabilise rapidement ; un *early-stopping* automatique permettrait d'accélérer encore ;
2. la présente phase SWD peut servir d'initialisation avant un éventuel affinage adversarial pour des applications qui l'exigeraient.

**Diversité accrue :** De plus la diversité des échantillons s'en trouve également accrue : en l'absence de discriminateur à patchs, le générateur n'est plus incité à coller fidèlement aux motifs locaux. On observe ainsi une meilleure capacité à produire des variations visuelles plausibles, sans recopie explicite des textures de l'image d'entrée.

**Limites de précision locale** Cependant, cette méthode introduit aussi une perte de précision locale. L'alignement des distributions de features ne garantit pas une reproduction fidèle des micro-détails : les bords peuvent être plus diffus, certaines textures plus uniformisées, et l'équilibre colorimétrique légèrement altéré (Figure 12). En d'autres termes, la fidélité structurelle est maintenue à moyen et large échelle, mais les motifs très fins sont parfois négligés, ce qui correspond aux limites naturelles de l'approche SWD dans un espace fixe non adapté.

## 5.7 Cas des textures

Dans ce cadre, l'utilisation de la SWD comme unique critère d'apprentissage s'est révélée particulièrement efficace. Les résultats obtenus présentent des motifs visuellement cohérents, variés et structurellement riches (Figure 13). En comparaison, les versions intégrant un discriminateur ont tendance à surajuster certains motifs ou à générer des artefacts perceptibles dans le cas de textures complexes.

Ce comportement s'explique par la nature même de la SWD, qui cherche à aligner les distributions de patchs entre les images générées et les images cibles. Dans le cas de textures sans structure globale imposée (herbe, mur, sable, etc.), cet objectif est particulièrement pertinent : il permet une génération fidèle au style sans contraintes de contenu global.

Ces résultats confirment que la SWD est particulièrement bien adaptée à la tâche de synthèse de texture pure. Contrairement aux métriques adversariales ou perceptuelles, elle capture efficacement les statistiques locales sans imposer de structure globale artificielle.

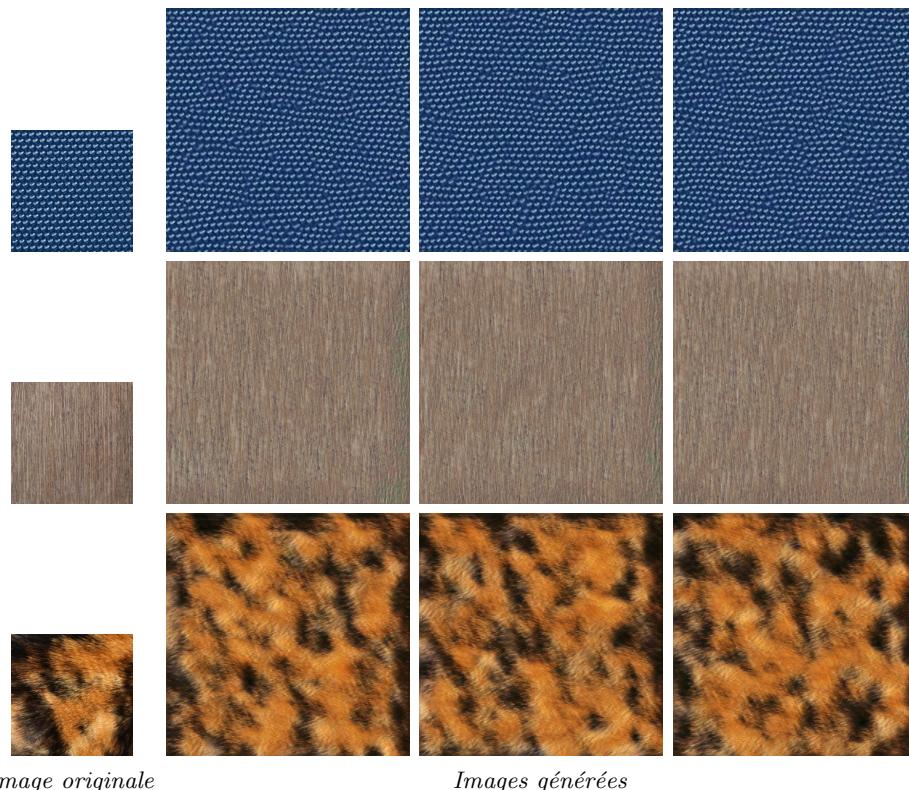


FIGURE 13 – Exemples de générations de texture. Première ligne : motif répété. Deuxième ligne : gaussien. Troisième ligne : non-gaussien

## 6 Comparaison des résultats

Nous comparons les résultats des trois variantes : l'implémentation originale du papier, notre version "vanilla" avec discriminateur, et notre version avec perte SWD (sans discriminateur). Cette comparaison porte à la fois sur des métriques quantitatives (*Wasserstein* et *LPIPS*) et qualitatives (échantillons visuels).

### 6.1 Résultats qualitatifs

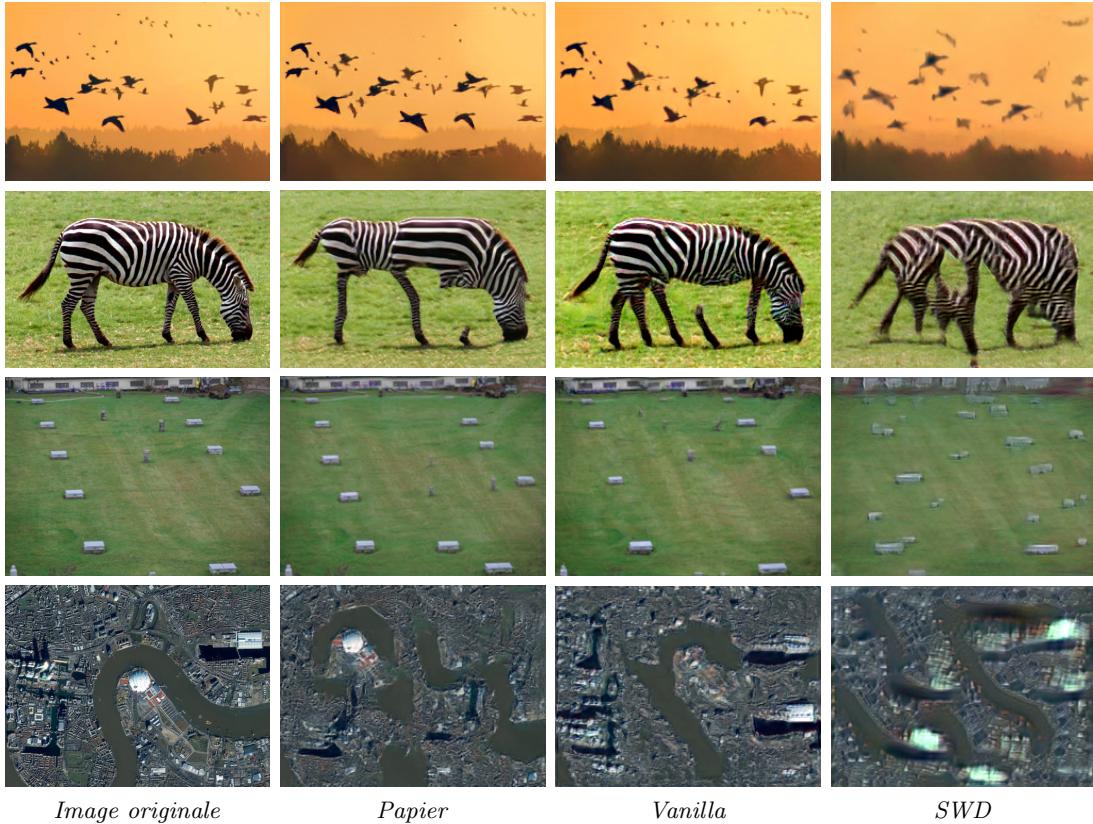


FIGURE 14 – Échantillons générés pour chaque image et chaque méthode. De haut en bas : Oiseaux, Zèbre, Herbe, Londres. De gauche à droite : image réelle, génération papier original, génération vanilla, génération SWD.

### 6.2 Résultats quantitatifs

Les deux tableaux suivants synthétisent les résultats moyens des distances Wasserstein et LPIPS pour chaque image et chaque méthode.

Image	Original	Vanilla	SWD
Herbe	26.47	<b>26.77</b>	28.27
Oiseaux	<b>26.05</b>	24.53	28.07
Londres	<b>33.91</b>	33.97	33.10
Zèbre	13.68	<b>12.16</b>	16.19

TABLE 5 – Comparaison des distances Wasserstein

<b>Image</b>	<b>Original</b>	<b>Vanilla</b>	<b>SWD</b>
Herbe	<b>0.323</b>	0.332	0.482
Oiseaux	0.373	<b>0.296</b>	0.412
Londres	<b>0.640</b>	0.646	0.647
Zèbre	0.472	<b>0.368</b>	0.605

TABLE 6 – Comparaison des distances LPIPS

Les tableaux détaillés de toutes les métriques sont disponibles en annexe.

### 6.3 Analyse

Notre version *vanilla* obtient des scores très proches de ceux de la version originale, ce qui valide la précision de notre réimplémentation. En particulier, pour l'image de zèbre, notre score LPIPS est significativement plus bas (0.368 contre 0.472 pour l'original), ce qui est cohérent avec les observations visuelles : nos échantillons présentent des motifs de rayures et une forme globale bien définis et proches de l'image de référence.

En utilisant uniquement la SWD, les distances Wasserstein restent du même ordre de grandeur (ex : 33.10 pour Londres contre 33.91 dans le papier), ce qui montre que le modèle conserve une certaine fidélité statistique.

Cependant, les scores LPIPS augmentent notablement (ex : 0.605 pour Zèbre, 0.482 pour Herbe), ce qui traduit une perte de cohérence visuelle. Les images générées respectent la distribution des textures, mais deviennent moins fidèles au contenu visuel initial. Ce résultat confirme l'intérêt d'utiliser LPIPS comme mesure de qualité perceptuelle.

Il est ainsi intéressant de regarder en particulier les différents résultats pour la synthèse de texture.

### 6.4 Cas des textures

Les différences entre méthodes sont ici particulièrement marquées (Figure 15). Visuellement, la version avec SWD produit des textures plus naturelles, diversifiées et localement cohérentes, sans artefacts répétitifs ni saturation excessive. Cela contraste avec la version originale, ainsi que notre version, qui tendent à générer des motifs surappris ou artificiellement régularisés.

Cette supériorité visuelle se reflète partiellement dans les scores quantitatifs, notamment pour les images de type texture (comme Herbe), où la SWD conserve des distances Wasserstein comparables tout en affichant des scores LPIPS plus élevés, traduisant une moindre similarité perceptuelle stricte, mais aussi une plus grande diversité visuelle.

Dans le cas de la synthèse de texture, la version SWD s'avère particulièrement adaptée. Elle permet une génération plus riche et fidèle aux motifs locaux, ce qui confirme l'intérêt de la SWD comme alternative efficace aux loss adversariales dans ce contexte particulier.

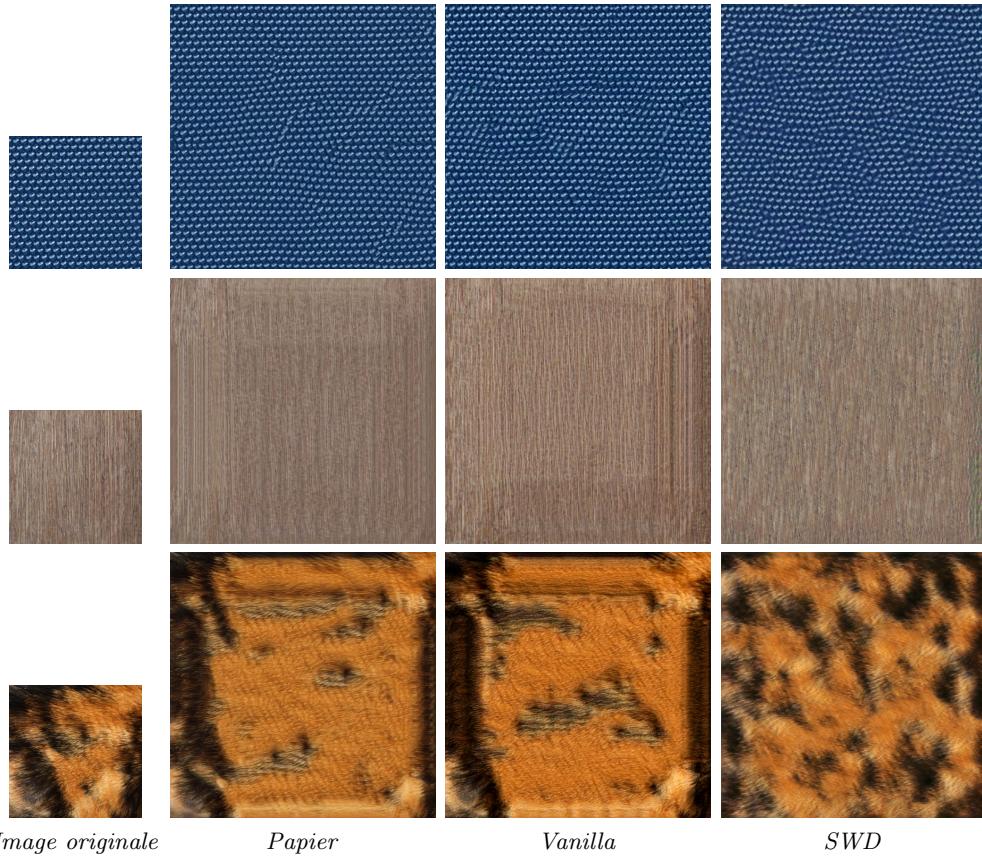


FIGURE 15 – Comparaison des générations de texture.

## 7 Conclusion

Dans ce travail, nous avons exploré différentes variantes de l'approche SinGAN pour la génération d'images à partir d'une seule. Après avoir reproduit les résultats du papier original, nous avons proposé deux alternatives : une réimplémentation *vanilla* fidèle mais allégée, puis une version sans discriminateur guidée uniquement par la *Sliced Wasserstein Distance* (SWD), mesurée dans l'espace des features d'Inception.

Nos résultats montrent que la version SWD permet de s'affranchir du couple générateur-discriminateur sans perte majeure de qualité statistique (Wasserstein), tout en apportant une plus grande diversité visuelle, notamment dans les cas de synthèse de texture. Toutefois, nous avons observé que la similarité perceptuelle directe (LPIPS) peut se dégrader, ce qui souligne une forme de décalage entre la distribution des patchs et la fidélité visuelle perçue, ce qui rend cette méthode en pratique non utilisable.

Une piste naturelle d'amélioration consisterait à combiner la SWD avec une perte LPIPS entre l'image générée et l'originale. Une telle combinaison pourrait permettre d'assurer à la fois une cohérence de distribution locale (via SWD) et une similarité perceptuelle globale (via LPIPS), apportant ainsi le meilleur des deux mondes.

Ainsi, si la suppression du discriminateur est envisageable dans certaines configurations, les résultats globaux confirment que le couple générateur-discriminateur conserve une meilleure capacité à modéliser fidèlement les contenus visuels complexes.

## Bibliographie

- [1] Tamar Rott SHAHAM, Tali DEKEL et Tomer MICHAELI. *SinGAN : Learning a Generative Model from a Single Natural Image*. 2019. arXiv : [1905.01164 \[cs.CV\]](https://arxiv.org/abs/1905.01164). URL : <https://arxiv.org/abs/1905.01164>.
- [2] Richard ZHANG et al. *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. 2018. arXiv : [1801.03924 \[cs.CV\]](https://arxiv.org/abs/1801.03924). URL : <https://arxiv.org/abs/1801.03924>.

## Annexe

Image	Moyenne	Écart-type	Min	Max
Herbe	26.469	1.038	23.940	28.574
Oiseaux	26.051	2.445	22.971	33.468
Londres	33.910	1.150	30.969	37.482
Zèbre	13.681	0.957	12.106	15.801

Image	Moyenne	Écart-type	Min	Max
Herbe	0.323	0.013	0.292	0.353
Oiseaux	0.373	0.021	0.315	0.418
Londres	0.640	0.012	0.607	0.664
Zèbre	0.472	0.033	0.378	0.541

TABLE 7 – Distances Wasserstein (haut) et LPIPS (bas) - papier original

Image	Moyenne	Écart-type	Min	Max
Birds	24.53	2.22	19.99	28.73
Zebra	12.16	0.62	10.80	13.29
Herbe	26.77	1.05	24.96	30.06
London	33.97	1.41	31.87	37.21

Image	Moyenne	Écart-type	Min	Max
Birds	0.296	0.018	0.258	0.328
Zebra	0.368	0.019	0.331	0.412
Herbe	0.332	0.014	0.307	0.363
London	0.646	0.010	0.614	0.666

TABLE 8 – Distances Wasserstein (haut) et LPIPS (bas) - notre implémentation vanilla

Image	SWD moyenne	Écart-type	Min	Max
Birds	28.07	1.48	25.00	31.71
Zebra	16.19	2.20	12.82	24.61
Herbe	28.27	1.20	25.80	31.32
London	33.10	1.78	29.86	37.92

Image	SWD moyenne	Écart-type	Min	Max
Birds	0.412	0.013	0.382	0.450
Zebra	0.605	0.022	0.542	0.641
Herbe	0.482	0.009	0.462	0.508
London	0.647	0.008	0.628	0.666

TABLE 9 – Distances Wasserstein (haut) et LPIPS (bas) - notre implémentation SWD