

# Méthodes d'apprentissage automatique pour la prise en compte du bruit dans les images de synthèse

Jérôme Buisine

8 décembre 2021

**Sous la direction de :**

Christophe Renaud (Directeur)  
Samuel Delepouille (Encadrant)



# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

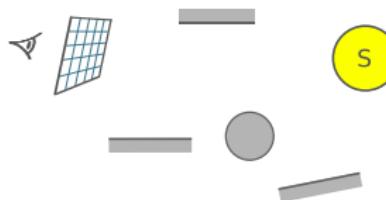
- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

## Contexte : Synthèse d'images

### Scène géométrique 3D

- Objets
- Caméra
- Sources de lumière



<sup>1</sup><https://github.com/mmp/pbrt-v4-scenes>

## Contexte : Synthèse d'images

### Scène géométrique 3D

- Objets
- Caméra
- Sources de lumière

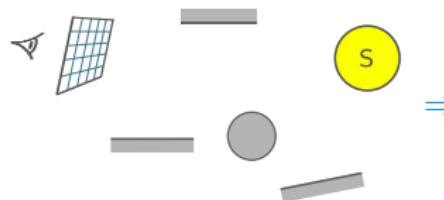


Image photo-réaliste<sup>1</sup>

<sup>1</sup><https://github.com/mmp/pbrt-v4-scenes>

## Contexte : Synthèse d'images

### Scène géométrique 3D

- Objets
- Caméra
- Sources de lumière

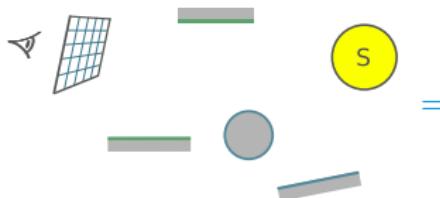
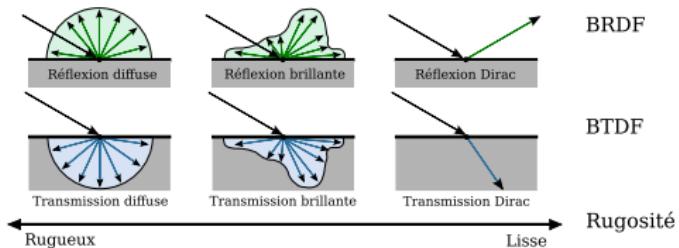


Image photo-réaliste<sup>1</sup>

### Transport de la lumière

- Propriétés des matériaux
- Interactions de la lumière

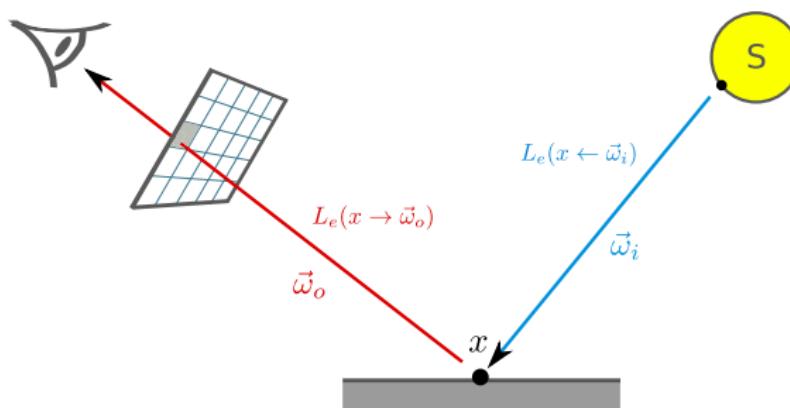


<sup>1</sup><https://github.com/mmp/pbrt-v4-scenes>

## Contexte : Illumination globale

Équation de rendu d'illumination globale [Kajiya, 1986]

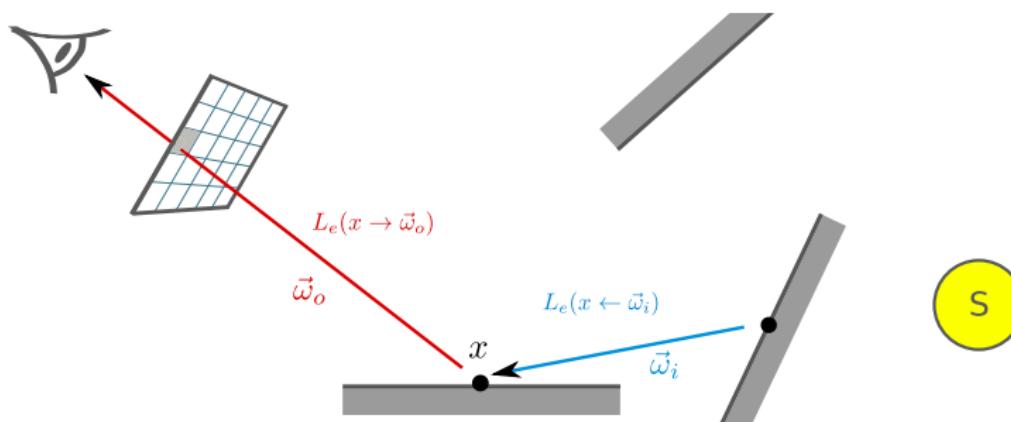
$$L(x \rightarrow \vec{\omega}_o) = L_e(x \rightarrow \vec{\omega}_o) + f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_e(x \leftarrow \vec{\omega}_i) \cos \theta_{\vec{\omega}_i}$$



## Contexte : Illumination globale

## Équation de rendu d'illumination globale [Kajiya, 1986]

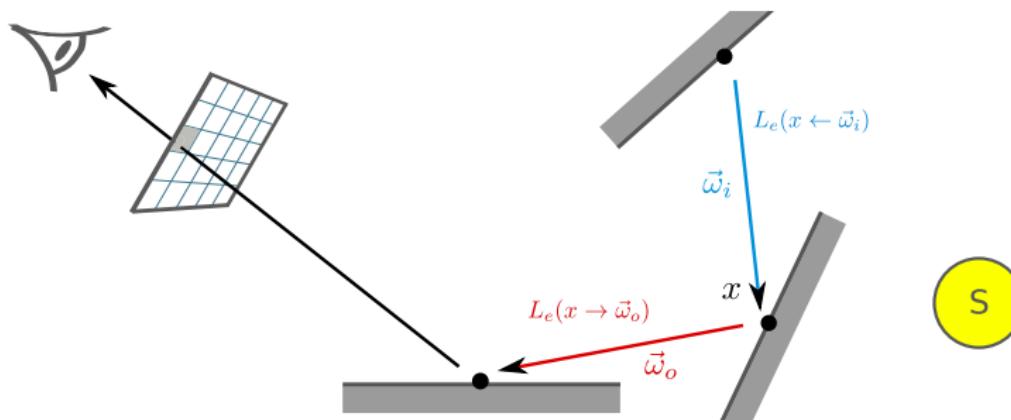
$$L(x \rightarrow \vec{\omega}_o) = L_e(x \rightarrow \vec{\omega}_o) + f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_e(x \leftarrow \vec{\omega}_i) \cos \theta_{\vec{\omega}_i}$$



## Contexte : Illumination globale

## Équation de rendu d'illumination globale [Kajiya, 1986]

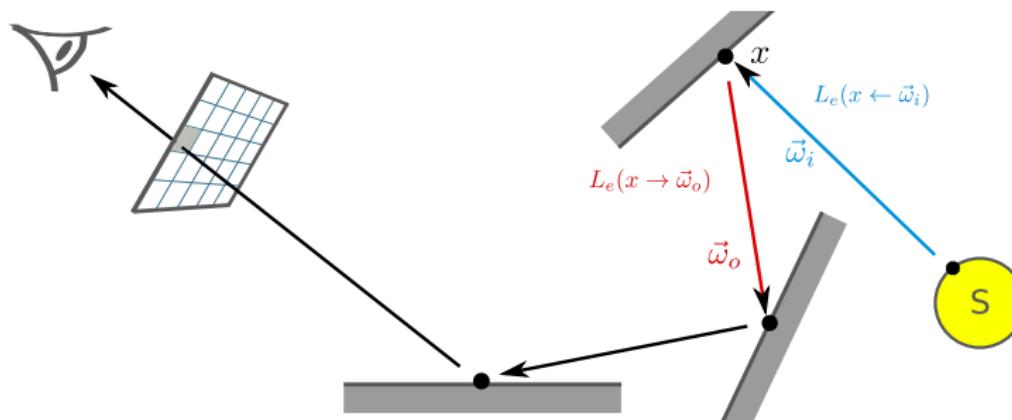
$$L(x \rightarrow \vec{\omega}_o) = L_e(x \rightarrow \vec{\omega}_o) + f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_e(x \leftarrow \vec{\omega}_i) \cos \theta_{\vec{\omega}_i}$$



## Contexte : Illumination globale

## Équation de rendu d'illumination globale [Kajiya, 1986]

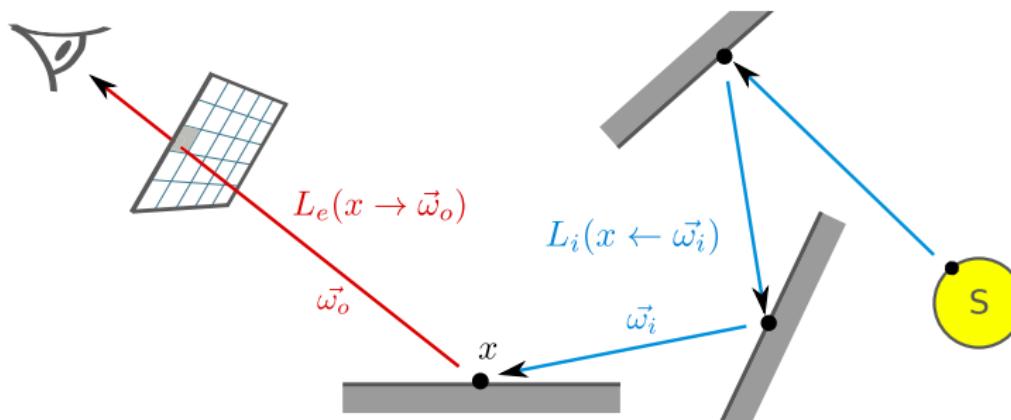
$$L(x \rightarrow \vec{\omega}_o) = L_e(x \rightarrow \vec{\omega}_o) + f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_e(x \leftarrow \vec{\omega}_i) \cos \theta_{\vec{\omega}_i}$$



## Contexte : Illumination globale

Équation de rendu d'illumination globale [Kajiya, 1986]

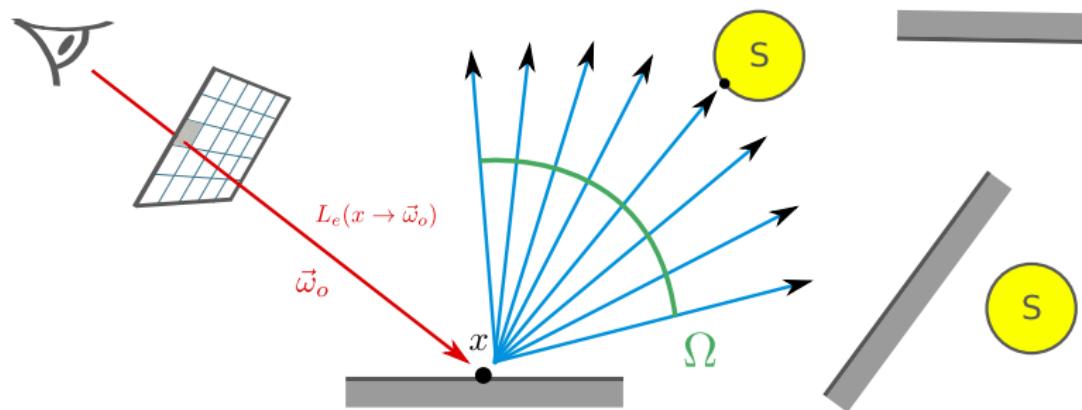
$$L(x \rightarrow \vec{\omega}_o) = L_e(x \rightarrow \vec{\omega}_o) + f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_i(x \leftarrow \vec{\omega}_i) \cos \theta_{\vec{\omega}_i}$$



## Contexte : Illumination globale

Équation de rendu d'illumination globale [Kajiya, 1986]

$$L(x \rightarrow \vec{\omega}_o) = L_e(x \rightarrow \vec{\omega}_o) + \int_{\Omega} f_r(x, \vec{\omega}_i \rightarrow \vec{\omega}_o) L_i(x \leftarrow \vec{\omega}_i) \cos \theta_{\vec{\omega}_i} d\omega_i$$



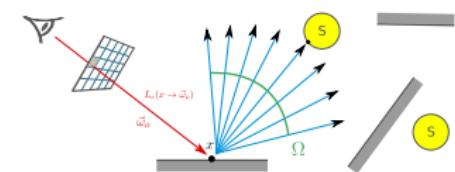
## Contexte : Méthode de Monte-Carlo

### Intégration par méthode de Monte-Carlo

$$F_N = \frac{1}{N} \sum_{j=1}^N \frac{f(x)}{p(x)} \approx \int_{\Omega} \frac{f(x)}{p(x)} p(x) dx$$

avec :

- $N$  le nombre d'échantillons ;
- $p$  une fonction de densité de probabilité.



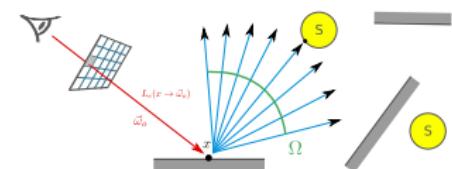
## Contexte : Méthode de Monte-Carlo

### Intégration par méthode de Monte-Carlo

$$F_N = \frac{1}{N} \sum_{j=1}^N \frac{f(x)}{p(x)} \approx \int_{\Omega} \frac{f(x)}{p(x)} p(x) dx$$

avec :

- $N$  le nombre d'échantillons ;
- $p$  une fonction de densité de probabilité.



### Remarques importantes

- Bruit visuel hautement perceptible
- Requiert un grand temps de calcul



## Contexte : Bruit résiduel perceptible Monte-Carlo



(a) 1 échantillon



(b) 20 échantillons



(c) 500 échantillons



(d) 10 000 échantillons

## Contexte : Bruit résiduel perceptible Monte-Carlo



(a) 1 échantillon



(b) 20 échantillons



(c) 500 échantillons

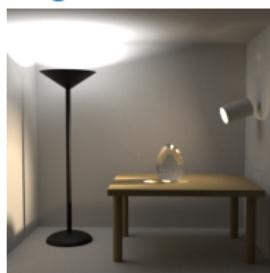


(d) 10 000 échantillons

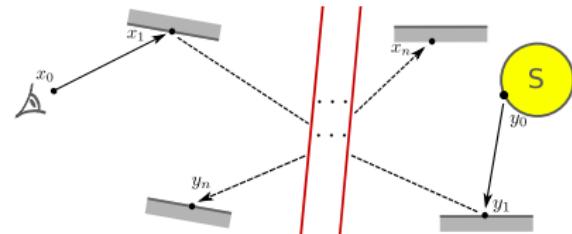
Rendu avec différents intégrateurs avec 1 000 échantillons par pixel :



(a) Tracé de chemins



(b) Bidirectionnel



Inconvénient du tracé de chemins bidirectionnel

- Difficilement adapté pour calculer sur carte GPU

## Contexte : Prise en compte du bruit

### Processus de post-traitement :

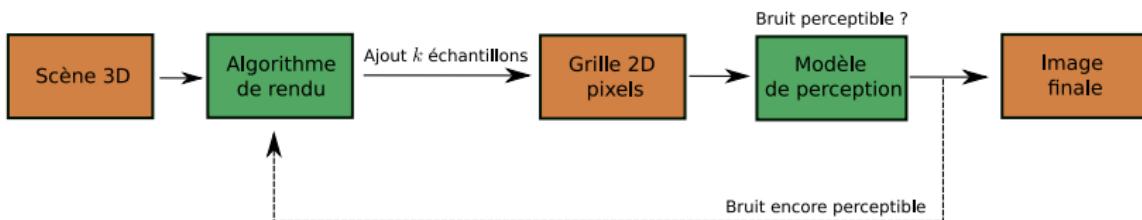


## Contexte : Prise en compte du bruit

### Processus de post-traitement :



### Seuil d'arrêt lors du rendu :



### Modèle de perception de bruit

- Modèles basés sur de l'apprentissage automatique

# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

## État de l'art : Concept de l'apprentissage automatique

### Type d'apprentissage

- Supervisé
- Non supervisé
- Semi-supervisé

## État de l'art : Concept de l'apprentissage automatique

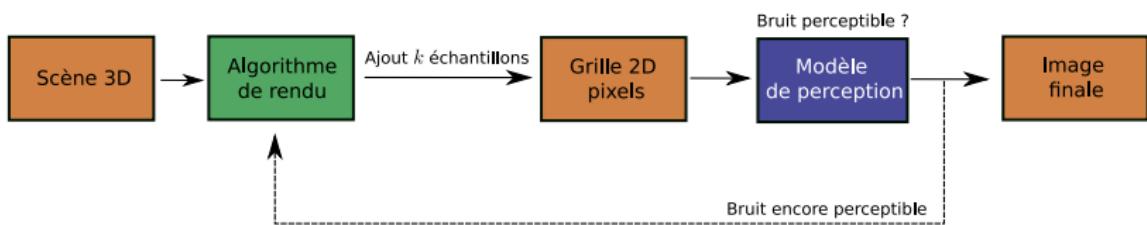
### Type d'apprentissage

- Supervisé
- Non supervisé
- Semi-supervisé

- $N$  paires de données :  $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$
- $y_i$  a été généré par une fonction inconnue  $y = f(x)$
- $h(x) \approx f(x)$

## État de l'art : Vers un modèle de perception

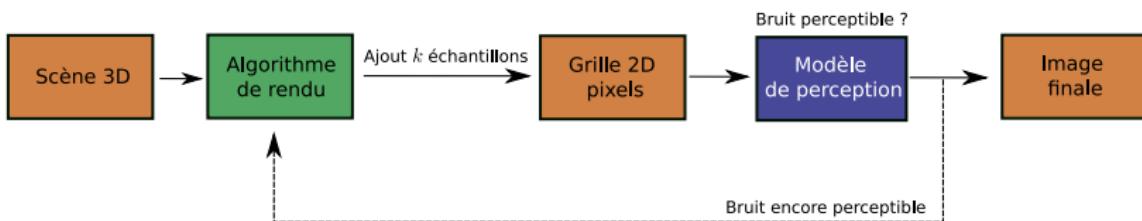
Interaction avec un modèle de détection de seuil :



<sup>1</sup>Support Vector Machine

# État de l'art : Vers un modèle de perception

Interaction avec un modèle de détection de seuil :



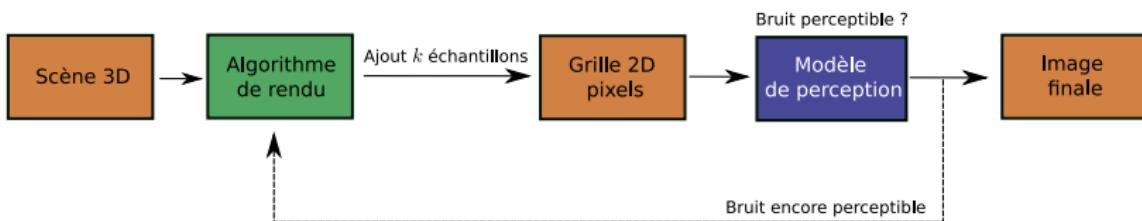
## Propositions de modèles de perception

- Modèle *Visual Discriminator Predictor* [Myszkowski, 1998]
- Méthode d'apprentissage Machine à Vecteur Support (SVM<sup>1</sup>)  
[Takouachet et al., 2017, Constantin et al., 2015, Constantin et al., 2016]

<sup>1</sup>Support Vector Machine

# État de l'art : Vers un modèle de perception

Interaction avec un modèle de détection de seuil :



## Propositions de modèles de perception

- Modèle *Visual Discriminator Predictor* [Myszkowski, 1998]
- Méthode d'apprentissage Machine à Vecteur Support (SVM<sup>1</sup>)  
[Takouachet et al., 2017, Constantin et al., 2015, Constantin et al., 2016]

<sup>1</sup>Support Vector Machine

## État de l'art : Vers un modèle de perception

### Base d'images de synthèse [Takouachet et al., 2017]

- 12 images de taille  $512 \times 512$
- Différents niveaux de bruit
- Images découpées en 16 blocs ( $16 \times 128 \times 128$ )

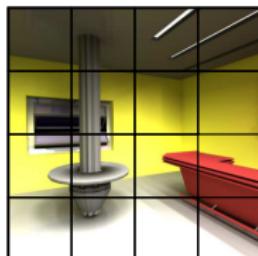
# État de l'art : Vers un modèle de perception

## Base d'images de synthèse [Takouachet et al., 2017]

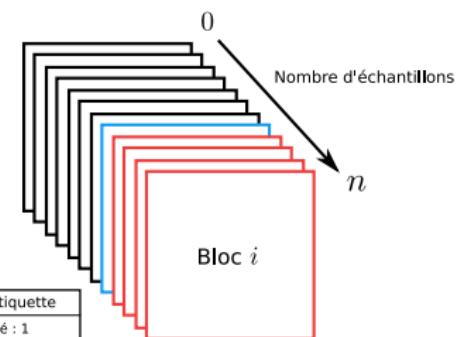
- 12 images de taille  $512 \times 512$
- Différents niveaux de bruit
- Images découpées en 16 blocs ( $16 \times 128 \times 128$ )

## Étiquettagement des données

- Récolte de seuil (nombre d'échantillons)
- Seuil de perception MOS
- Tâche de classification binaire

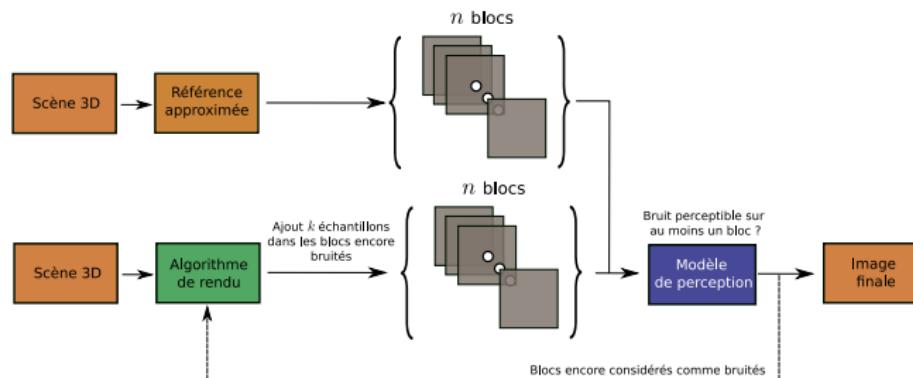


Valeur d'étiquette
— Bruité : 1
— Seuil : 0
— Non bruité : 0



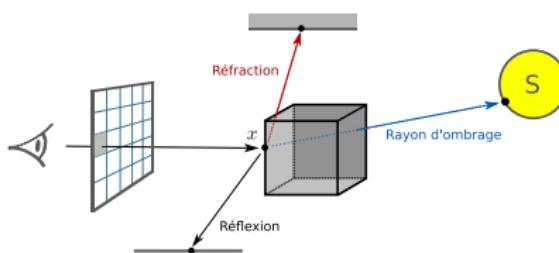
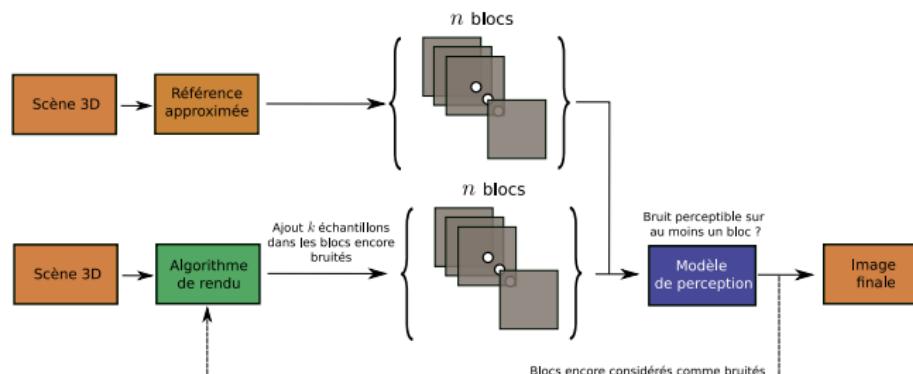
# État de l'art : Vers un modèle de perception

## Interaction avec un modèle de perception



# État de l'art : Vers un modèle de perception

## Interaction avec un modèle de perception



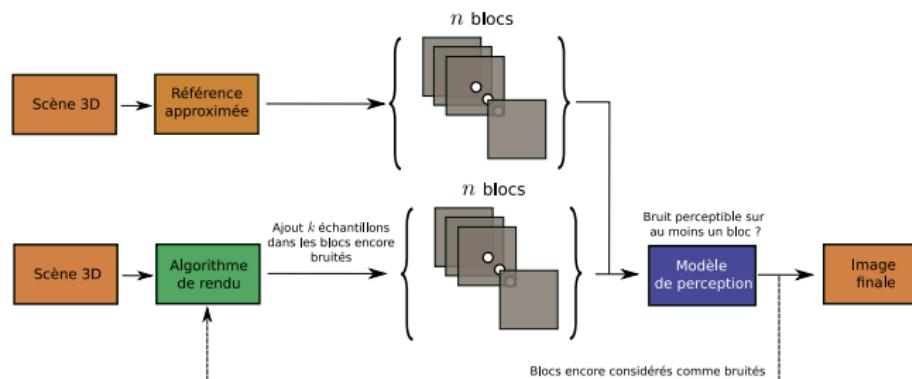
Référence approximée

## Espace de couleur CIE $L^* a^* b^*$

- Extraction de la composante  $L$  (1 canal)
- Bruit semble affecter essentiellement  $L$  [Carne et al., 2008]

# État de l'art : Vers un modèle de perception

## Interaction avec un modèle de perception



### Entrée du modèle SVM relativement à une référence approximée

- Deux masques de bruit de taille  $128 \times 128$  [Takouachet et al., 2017]
- 26 attributs extraits de la différence des images de taille  $128 \times 128$  [Constantin et al., 2015]
- Masque de bruit de taille  $128 \times 128$  [Constantin et al., 2016]

## État de l'art : Vers un modèle de perception

### Référence approximée

- Ne prend pas en compte certains effets lumineux :
  - Caustiques
  - L'éclairage global
  - La dispersion lumineuse

# État de l'art : Vers un modèle de perception

## Référence approximée

- Ne prend pas en compte certains effets lumineux :
  - Caustiques
  - L'éclairage global
  - La dispersion lumineuse

## Base d'images

- Scènes avec peu d'effets lumineux complexes
- Manque de diversité des scènes



# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

### ■ Base d'images de synthèse

- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

## Base d'images : Génération des données

### Génération d'images

- 40 images de taille  $1920 \times 1080$  (28 scènes)
- $\approx 10$  To (10 000 images intermédiaires <sup>1</sup>)
- Tracé de chemins avec PBRT-v3 [Pharr et al., 2016]

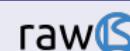
<sup>1</sup>Images calculées à partir de la plateforme CALCULCO, gérée par le SCoSI de l'ULCO

## Base d'images : Génération des données

### Génération d'images

- 40 images de taille  $1920 \times 1080$  (28 scènes)
- $\approx 10$  To (10 000 images intermédiaires <sup>1</sup>)
- Tracé de chemins avec PBRT-v3 [Pharr et al., 2016]

### Format RAW Light Simulation



- format RAW
- Metadata de génération

<sup>1</sup>Images calculées à partir de la plateforme CALCULCO, gérée par le SCoSI de l'ULCO

## Base d'images : Génération des données

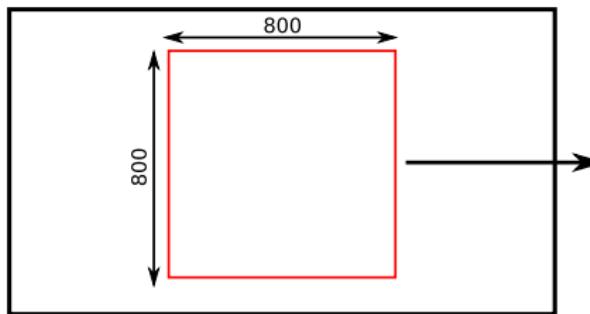
### Génération d'images

- 40 images de taille  $1920 \times 1080$  (28 scènes)
- $\approx 10$  To (10 000 images intermédiaires<sup>1</sup>)
- Tracé de chemins avec PBRT-v3 [Pharr et al., 2016]

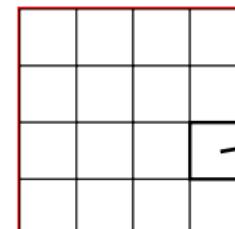
### Format RAW Light Simulation



- format RAW
- Metadata de génération



### Division en blocs



200 x 200

Image de scène ( $1920 \times 1080$ )

<sup>1</sup>Images calculées à partir de la plateforme CALCULCO, gérée par le SCoSI de l'ULCO

## Base d'images : Paramètres

### Base d'images de synthèse [Takouachet et al., 2017]

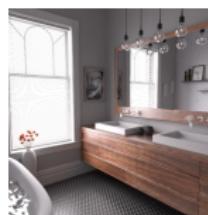
- **12 images de taille  $512 \times 512$  (7 scènes)**
- Différents niveaux de bruit de 100 à 10 000 échantillons par pixels
- Pas de **100** échantillons entre deux images successives
- Images découpées en 16 blocs ( $16 \times 128 \times 128$ )

### Nouvelle base d'images

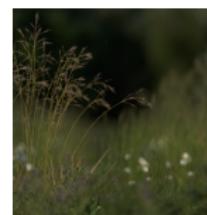
- **40 images de taille  $800 \times 800$  (28 scènes)**
- Différents niveaux de bruit de 20 à 10 000 échantillons par pixel
- Pas de **20** échantillons entre deux images successives
- Images découpées en 16 blocs ( $16 \times 200 \times 200$ )



(a) Glass



(b) Bathroom



(c) Landscape



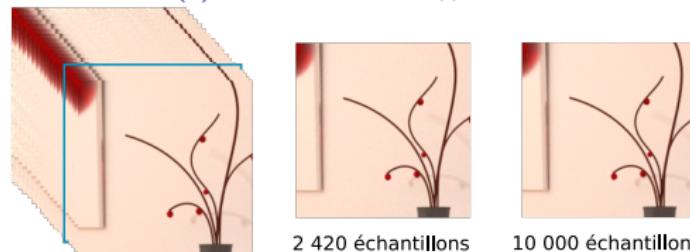
(d) San-Miguel

## Base d'images : Application de récolte de seuils

Acquisition des seuils auprès d'une population experte :



(a) Vue d'ensemble de l'application

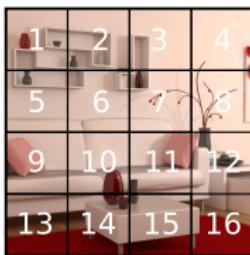


2 420 échantillons      10 000 échantillons

(b) Aperçu de l'interaction possible du participant sur un bloc

## Base d'images : Données de référence

Comparaisons avec la *Structural Similarity Index Measure (SSIM)* [Wang et al., 2004] :



(a) Numérotation des blocs d'images

8680	5286	6553	2293
7153	7086	6420	2626
4420	6580	4560	7820
7086	8286	6286	6693

(b) Seuils subjectifs (MOS)



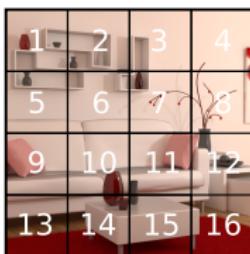
(c) Image reconstruite  
SSIM: 0.9903



(d) Référence  
SSIM: 1.0

## Base d'images : Données de référence

Comparaisons avec la *Structural Similarity Index Measure (SSIM)* [Wang et al., 2004] :



(a) Numérotation des blocs d'images

8680	5286	6553	2293
7153	7086	6420	2626
4420	6580	4560	7820
7086	8286	6286	6693

(b) Seuils subjectifs (MOS)



(c) Image reconstruite  
SSIM: 0.9903



(d) Référence  
SSIM: 1.0

## Base d'images : Données de référence

Comparaisons avec la *Structural Similarity Index Measure (SSIM)* [Wang et al., 2004] :

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

(a) Numérotation des blocs d'images

8680	5286	6553	2293
7153	7086	6420	2626
4420	6580	4560	7820
7086	8286	6286	6693

(b) Seuils subjectifs (MOS)



(c) Image reconstruite  
SSIM: 0.9903



(d) Référence  
SSIM: 1.0

### Base d'images

Publiée sur la plateforme Zenodo [Buisine et al., 2021d]

# Plan

## 1 Contexte

## 2 État de l'art

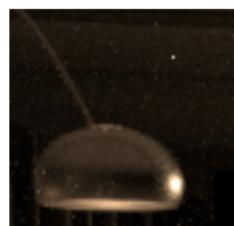
## 3 Contributions

- Base d'images de synthèse
- **Gestion des valeurs aberrantes lors du rendu**
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

## Valeurs aberrantes : Bruit spécifique généré en sortie du rendu

10 000  
échantillons



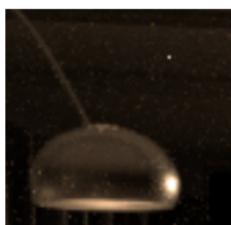
(a) Tracé de chemins



(b) Tracé de chemins

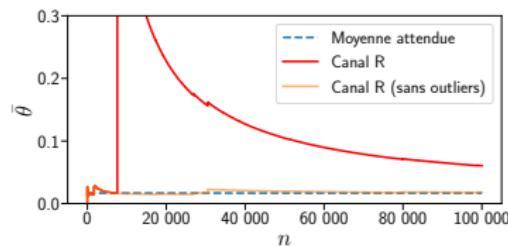
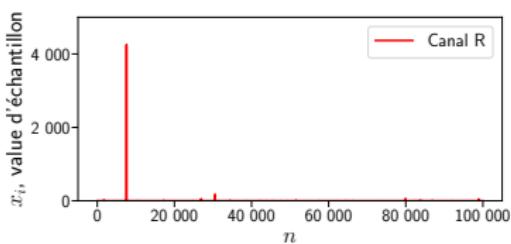
## Valeurs aberrantes : Bruit spécifique généré en sortie du rendu

10 000  
échantillons



(a) Tracé de chemins

(b) Tracé de chemins



## Valeurs aberrantes : Un problème de *firefly*

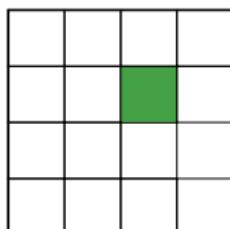
### Travaux relatifs à la gestion des *fireflies*

- Identification et suppression [DeCoro et al., 2010]
- Estimateur de la médiane des moyennes (MoN) [Jung et al., 2015]
- Pondération adaptée des valeurs aberrantes [Zirr et al., 2018]

## Valeurs aberrantes : Un problème de *firefly*

### Travaux relatifs à la gestion des *fireflies*

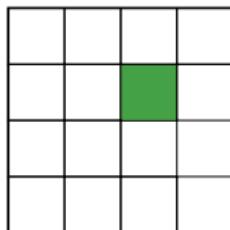
- Identification et suppression [DeCoro et al., 2010]
- Estimateur de la médiane des moyennes (MoN) [Jung et al., 2015]
- Pondération adaptée des valeurs aberrantes [Zirr et al., 2018]



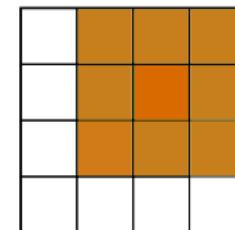
## Valeurs aberrantes : Un problème de *firefly*

### Travaux relatifs à la gestion des *fireflies*

- Identification et suppression [DeCoro et al., 2010]
- Estimateur de la médiane des moyennes (MoN) [Jung et al., 2015]
- Pondération adaptée des valeurs aberrantes [Zirr et al., 2018]



Informations locales



Informations non locales

## Valeurs aberrantes : Médiane des moyennes

L'estimateur MoN [Blair, 1985, Jerrum et al., 1986] :

$$\hat{\mu}_{MoN} = \text{médiane}\left(\frac{1}{k} \sum_{i=1}^k x_i, \dots, \frac{1}{k} \sum_{i=n-k+1}^n x_i\right)$$

avec des ajustements mineurs si  $\frac{n}{k}$  n'est pas un nombre entier.

## Valeurs aberrantes : Médiane des moyennes

L'estimateur MoN [Blair, 1985, Jerrum et al., 1986] :

$$\hat{\mu}_{MoN} = \text{médiane}\left(\frac{1}{k} \sum_{i=1}^k x_i, \dots, \frac{1}{k} \sum_{i=n-k+1}^n x_i\right)$$

avec des ajustements mineurs si  $\frac{n}{k}$  n'est pas un nombre entier.

---

### Propriétés de la médiane [Hampel, 1971]

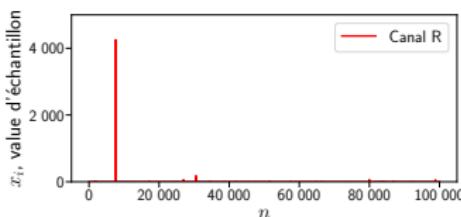
- Plus robuste que la moyenne
- Implique une convergence plus longue que la moyenne

### Inconvénient de MoN

- N'exploite pas assez les informations autres que la médiane [Orenstein, 2019]

# Valeurs aberrantes : Identification du meilleur estimateur

## Comment et quand choisir le meilleur estimateur ?



### Moyenne

$$\hat{\theta} = \frac{1}{n} \sum_{i=1}^n$$

### MoN



### Caractéristiques

- Fidélité
- Peu robuste

### Caractéristiques

- Robustesse
- Convergence plus longue
- Informations supplémentaires

## Valeurs aberrantes : Vers l'identification des valeurs aberrantes

### Vers l'utilisation du coefficient de Gini :

$$G = \frac{\frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n |x_i - x_j|}{2 \times \left( \frac{1}{n} \sum_{k=1}^n x_k \right)}$$

#### Coefficient de Gini

- Utilisé en économétrie
- Met en évidence les inégalités sociales

## Valeurs aberrantes : Vers l'identification des valeurs aberrantes

### Vers l'utilisation du coefficient de Gini :

$$G = \frac{\frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n |x_i - x_j|}{2 \times \left( \frac{1}{n} \sum_{k=1}^n x_k \right)}$$

$$G = \frac{2 \sum_{j=1}^M j \hat{\theta}_j}{M \sum_{j=1}^M \hat{\theta}_j} - \frac{M+1}{M}$$

Adaptée à l'estimateur de MoN [Damgaard, 2003]

#### Coefficient de Gini

- Utilisé en économétrie
- Met en évidence les inégalités sociales

- $M$  moyennes calculées
- $\hat{\theta}_j$  avec  $j \in [1, M]$
- Indexées par ordre croissant ( $\hat{\theta}_j \leq \hat{\theta}_{j+1}$ )

## Valeurs aberrantes : Vers l'identification des valeurs aberrantes

### Vers l'utilisation du coefficient de Gini :

$$G = \frac{\frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n |x_i - x_j|}{2 \times \left( \frac{1}{n} \sum_{k=1}^n x_k \right)}$$

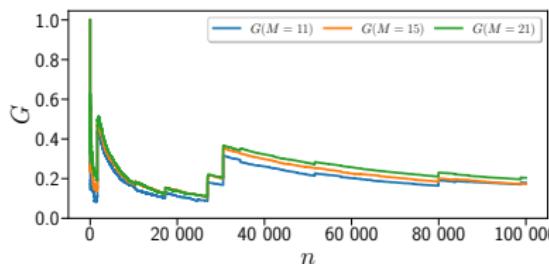
$$G = \frac{2 \sum_{j=1}^M j \hat{\theta}_j}{M \sum_{j=1}^M \hat{\theta}_j} - \frac{M+1}{M}$$

Adaptée à l'estimateur de MoN [Damgaard, 2003]

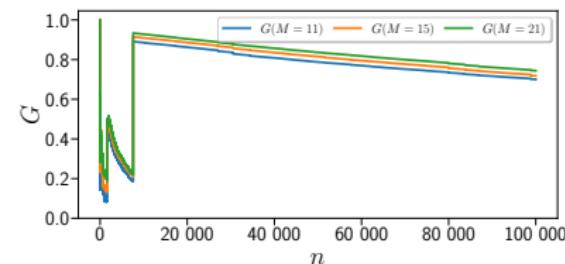
#### Coefficient de Gini

- Utilisé en économétrie
- Met en évidence les inégalités sociales

- $M$  moyennes calculées
- $\hat{\theta}_j$  avec  $j \in [1, M]$
- Indexées par ordre croissant ( $\hat{\theta}_j \leq \hat{\theta}_{j+1}$ )



(a) Sans valeur aberrante



(b) Avec valeur aberrante

## Valeurs aberrantes : Estimateurs proposés

### Sélection de l'estimateur

#### Estimateur binaire

$$G\text{-MoN}_b = \begin{cases} \bar{\theta} & \text{si } G \leq 0.25, \\ \hat{\mu}_{MoN} & \text{sinon} \end{cases}$$

### Informations supplémentaires

#### Estimateur adaptatif

$$G\text{-MoN} = \frac{\sum_{j=1+c}^{M-c} \hat{\theta}_j}{M - 2c}$$

## Valeurs aberrantes : Estimateurs proposés

### Sélection de l'estimateur

#### Estimateur binaire

$$G\text{-MoN}_b = \begin{cases} \bar{\theta} & \text{si } G \leq 0.25, \\ \hat{\mu}_{MoN} & \text{sinon} \end{cases}$$

### Informations supplémentaires

#### Estimateur adaptatif

$$G\text{-MoN} = \frac{\sum_{j=1+c}^{M-c} \hat{\theta}_j}{M - 2c}$$

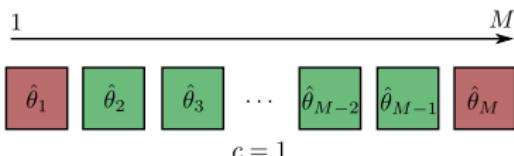


Figure: Équation MoN adaptatif pour  $c = 1$

## Valeurs aberrantes : Estimateurs proposés

### Sélection de l'estimateur

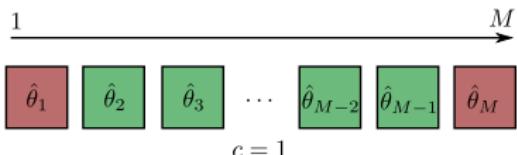
#### Estimateur binaire

$$G\text{-MoN}_b = \begin{cases} \bar{\theta} & \text{si } G \leq 0.25, \\ \hat{\mu}_{MoN} & \text{sinon} \end{cases}$$

### Informations supplémentaires

#### Estimateur adaptatif

$$G\text{-MoN} = \frac{\sum_{j=1+c}^{M-c} \hat{\theta}_j}{M - 2c}$$



#### Paramètre automatique $c$

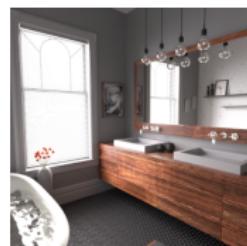
$$c = \lfloor G \times k \rfloor \text{ avec } k = \lfloor \frac{M}{2} \rfloor$$

Figure: Équation MoN adaptatif pour  $c = 1$

## Valeurs aberrantes : Protocole de comparaison

Base de 4 images de références :

Sans firefly  
(tracé de chemins)



(a) Bathroom

Avec fireflies  
(bidirectionnel)

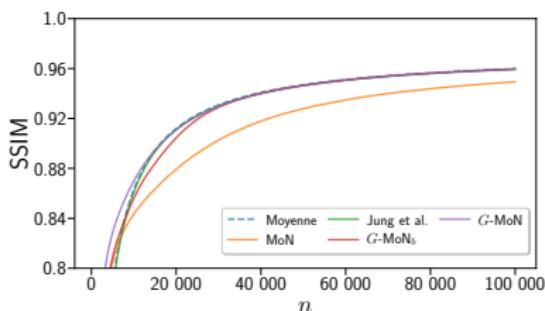


(c) Veach

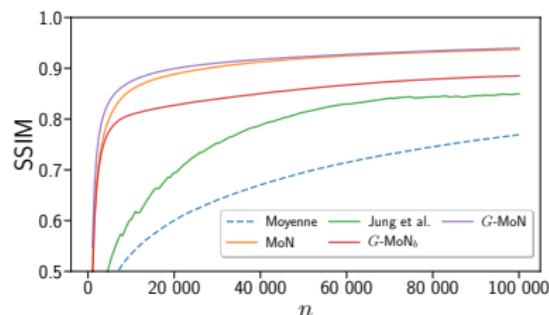


(d) Villa

## Valeurs aberrantes : Estimateurs locaux



(a) Bathroom (sans firefly)

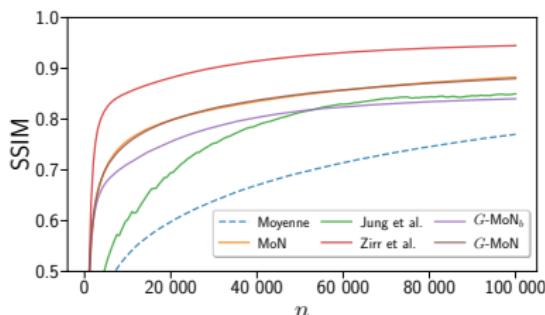
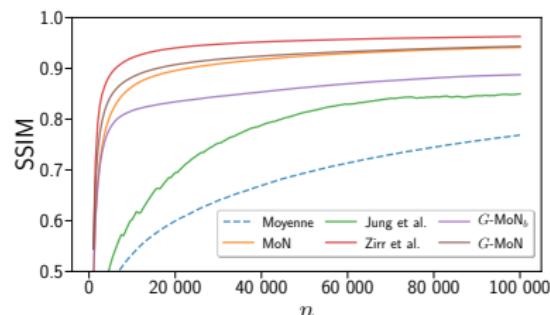


(b) Veach (avec fireflies)

### Paramètres

- Estimateurs basés sur  $G$  avec  $M = 21$
- MoN avec  $M = 21$
- Jung et al. avec paramètre automatique

## Valeurs aberrantes : Estimateurs non locaux

(a) Veach ( $M = 5$ )(b) Veach ( $M = 25$ )

### Paramètres

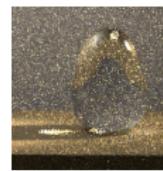
- Estimateurs basés sur  $G$  et MoN avec  $M \in \{5, 25\}$
- Zirr et al.

## Valeurs aberrantes : Interprétation des résultats

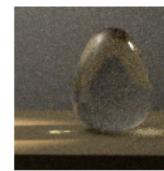
### Avantages de l'estimateur adaptatif G-MoN

- Basé uniquement sur les informations locales
- Déetecte la présence ou non de valeurs aberrantes
- Permet l'exploitation du compromis médiane et moyenne (robustesse et fidélité)

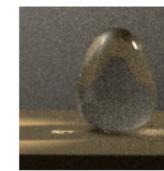
500  
échantillons



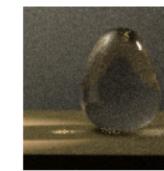
(a) Moyenne



(b) MoN



(c) G-MoN



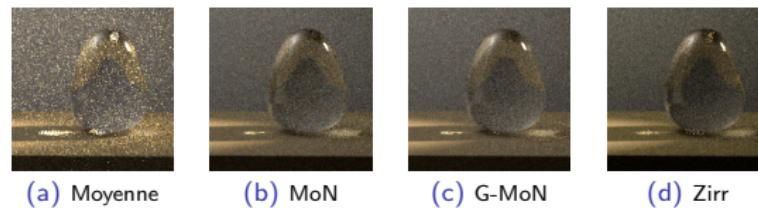
(d) Zirr

## Valeurs aberrantes : Interprétation des résultats

### Avantages de l'estimateur adaptatif G-MoN

- Basé uniquement sur les informations locales
- Déetecte la présence ou non de valeurs aberrantes
- Permet l'exploitation du compromis médiane et moyenne (robustesse et fidélité)

500  
échantillons



### Inconvénients de l'estimateur adaptatif G-MoN

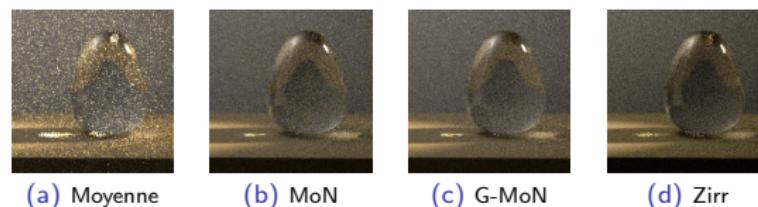
- Limité aux informations locales (pour le moment)
- Moins bon que l'estimateur non-local de [Zirr et al., 2018]
- Peut posséder toujours un risque de présence de fireflies si  $M$  petit

## Valeurs aberrantes : Interprétation des résultats

### Avantages de l'estimateur adaptatif G-MoN

- Basé uniquement sur les informations locales
- Déetecte la présence ou non de valeurs aberrantes
- Permet l'exploitation du compromis médiane et moyenne (robustesse et fidélité)

500  
échantillons



### Inconvénients de l'estimateur adaptatif G-MoN

- Limité aux informations locales (pour le moment)
- Moins bon que l'estimateur non-local de [Zirr et al., 2018]
- Peut posséder toujours un risque de présence de fireflies si  $M$  petit

### Publication

Conférence *Eurographics Symposium on Rendering* [Buisine et al., 2021b]

# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit**
- Sélection d'attributs comme un problème d'optimisation

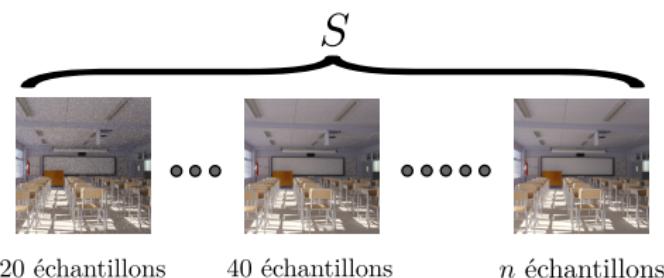
## 4 Conclusion et perspectives

## Modèle de perception

### Modèle de perception - Réseau de neurones récurrent (RNN)

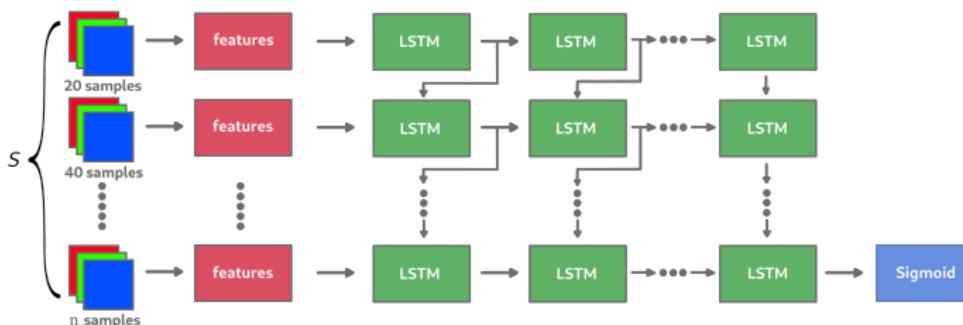
## Modèle de perception RNN : Méthode proposée

### Vers l'utilisation d'une fenêtre glissante



## Modèle de perception RNN : Méthode proposée

Modèle RNN avec des cellules *Long Short Term Memory* [Hochreiter and Schmidhuber, 1997] :



### Paramètres de l'architecture

- $S$  la taille de la fenêtre glissante
- 3 couches de cellules LSTM : 512, 128 et 32

# Modèle de perception RNN : L'entropie SVD

## Décomposition en valeurs singulières (SVD)

$$L = U \cdot \Sigma \cdot V^T$$

### Propriétés de la SVD

- $U$  de taille  $M \times M$
- $V^T$  de taille  $N \times N$
- Éléments diagonaux de  $\Sigma$  sont les valeurs singulières de  $L$

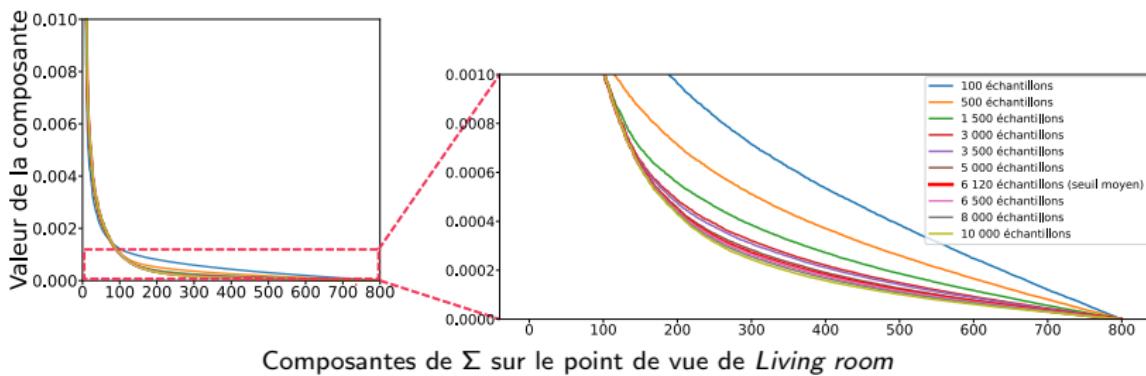
# Modèle de perception RNN : L'entropie SVD

## Décomposition en valeurs singulières (SVD)

$$L = U \cdot \Sigma \cdot V^T$$

### Propriétés de la SVD

- $U$  de taille  $M \times M$
- $V^T$  de taille  $N \times N$
- Éléments diagonaux de  $\Sigma$  sont les valeurs singulières de  $L$



# Modèle de perception RNN : L'entropie SVD

## Décomposition en valeurs singulières (SVD)

$$L = U \cdot \Sigma \cdot V^T$$

### Propriétés de la SVD

- $U$  de taille  $M \times M$
- $V^T$  de taille  $N \times N$
- Éléments diagonaux de  $\Sigma$  sont les valeurs singulières de  $L$

Après reconstruction [Liu, 2014] :



(a) Canal L (500 échantillons)



(b) Composantes 1 à 50

# Modèle de perception RNN : L'entropie SVD

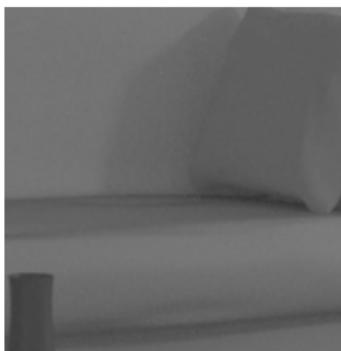
## Décomposition en valeurs singulières (SVD)

$$L = U \cdot \Sigma \cdot V^T$$

### Propriétés de la SVD

- $U$  de taille  $M \times M$
- $V^T$  de taille  $N \times N$
- Éléments diagonaux de  $\Sigma$  sont les valeurs singulières de  $L$

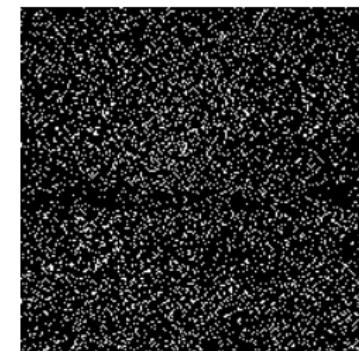
Après reconstruction [Liu, 2014] :



(a) Canal L (500 échantillons)



(b) Composantes 1 à 50



(c) Composantes 51 à 200

# Modèle de perception RNN : L'entropie SVD

**Mesure de l'entropie SVD** [Banerjee and Pal, 2014] :

Importance relative des composantes de  $\Sigma$

$$\bar{\sigma}_i = \sigma_i^2 / \sum_{p=1}^O \sigma_p^2$$

avec  $O$  le rang de la matrice  $L$ .

L'entropie SVD basée sur l'entropie de Shannon

$$H_{SVD} = -\frac{1}{\log(O)} \sum_{i=1}^O \bar{\sigma}_i \log(\bar{\sigma}_i)$$

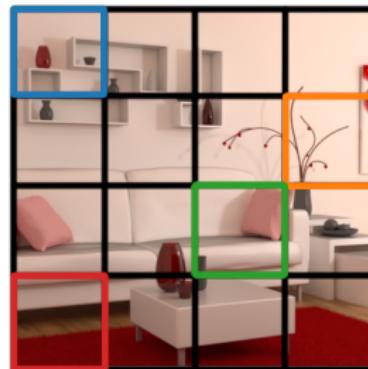
# Modèle de perception RNN : L'entropie SVD

**Mesure de l'entropie SVD** [Banerjee and Pal, 2014] :

Importance relative des composantes de  $\Sigma$

$$\bar{\sigma}_i = \sigma_i^2 / \sum_{p=1}^O \sigma_p^2$$

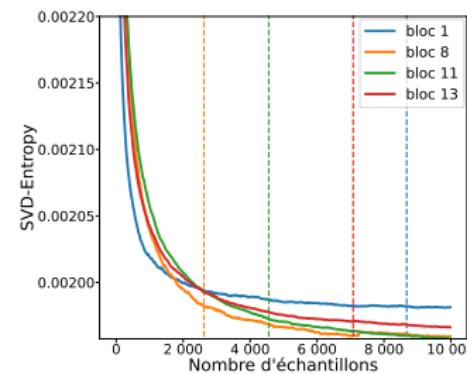
avec  $O$  le rang de la matrice  $L$ .



(a) Blocs sélectionnés

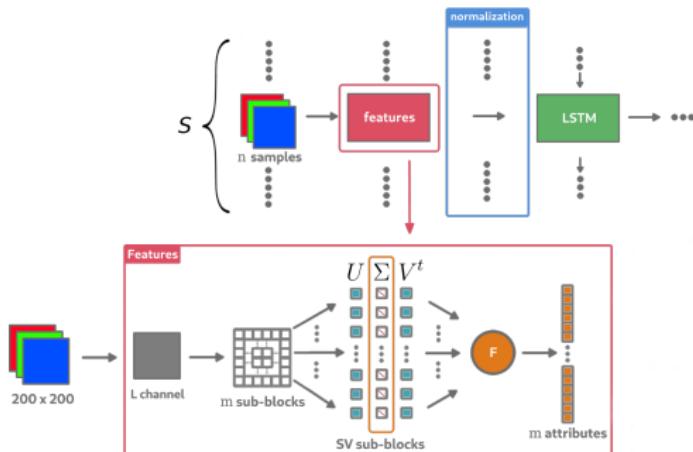
L'entropie SVD basée sur l'entropie de Shannon

$$H_{SVD} = -\frac{1}{\log(O)} \sum_{i=1}^O \bar{\sigma}_i \log(\bar{\sigma}_i)$$



(b)  $H_{SVD}$  (normalisées)

# Modèle de perception RNN : Méthode proposée



avec  $F \in [H_{SVD}, H_{SVD}^1, H_{SVD}^2]$

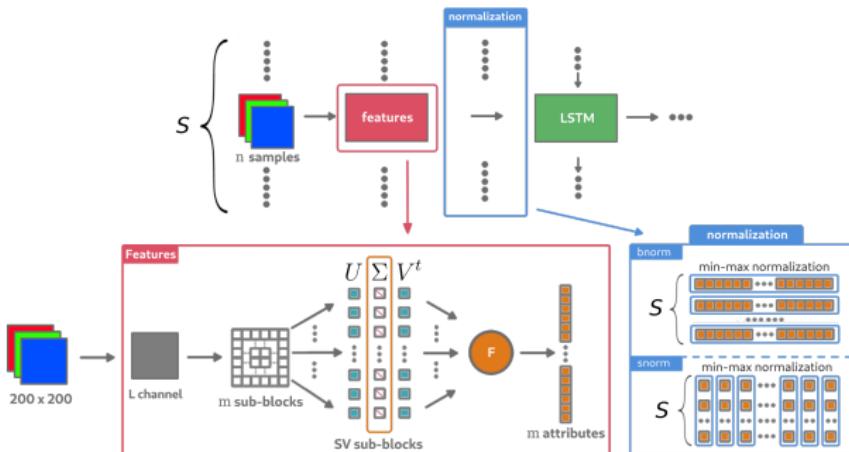
## Premières composantes de $\Sigma$

$$H_{SVD}^1 = -\frac{1}{\log(\frac{O}{4})} \sum_{i=0}^{O/4} \bar{\sigma}_i \log_2 \bar{\sigma}_i$$

## Dernières composantes de $\Sigma$

$$H_{SVD}^2 = -\frac{1}{\log(O - \frac{O}{4})} \sum_{i=O/4}^O \bar{\sigma}_i \log_2 \bar{\sigma}_i$$

# Modèle de perception RNN : Méthode proposée



avec  $F \in [H_{SVD}, H_{SVD}^1, H_{SVD}^2]$

## Premières composantes de $\Sigma$

$$H_{SVD}^1 = -\frac{1}{\log(\frac{O}{4})} \sum_{i=0}^{O/4} \bar{\sigma}_i \log_2 \bar{\sigma}_i$$

## Dernières composantes de $\Sigma$

$$H_{SVD}^2 = -\frac{1}{\log(O - \frac{O}{4})} \sum_{i=O/4}^O \bar{\sigma}_i \log_2 \bar{\sigma}_i$$

# Modèle de perception RNN : Paramètres étudiés

## Jeux de paramètres

- $S \in [3, 4, \dots, 10]$  ;
- $m$  de sous-blocs par bloc d'image, avec  $m \in [4, 25, 100, 400]$
- Le pas d'échantillonnage entre les images  $n \in [20, 40, 80]$
- Méthode de normalisation
- Valeur SVD Entropie ( $F \in [H_{SVD}, H_{SVD}^1, H_{SVD}^2]$ )

# Modèle de perception RNN : Paramètres étudiés

## Jeux de paramètres

- $S \in [3, 4, \dots, 10]$  ;
- $m$  de sous-blocs par bloc d'image, avec  $m \in [4, 25, 100, 400]$
- Le pas d'échantillonnage entre les images  $n \in [20, 40, 80]$
- Méthode de normalisation
- Valeur SVD Entropie ( $F \in [H_{SVD}, H_{SVD}^1, H_{SVD}^2]$ )

## Apprentissage et comparaison

- 30 époques
- 12 blocs sélectionnés pour chaque image en apprentissage
- Comparaison avec le score **AUC ROC** sur la base de test

# Modèle de perception RNN : Paramètres étudiés

## Jeux de paramètres

- $S \in [3, 4, \dots, 10]$  ;
- $m$  de sous-blocs par bloc d'image, avec  $m \in [4, 25, 100, 400]$
- Le pas d'échantillonnage entre les images  $n \in [20, 40, 80]$
- Méthode de normalisation
- Valeur SVD Entropie ( $F \in [H_{SVD}, H_{SVD}^1, H_{SVD}^2]$ )

## Apprentissage et comparaison

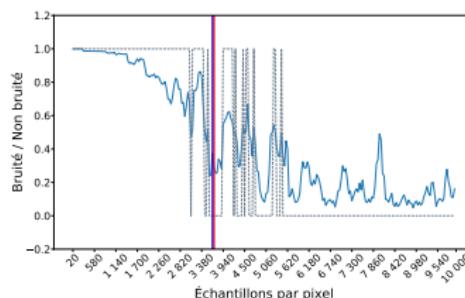
- 30 époques
- 12 blocs sélectionnés pour chaque image en apprentissage
- Comparaison avec le score **AUC ROC** sur la base de test

## Meilleur modèle retenu

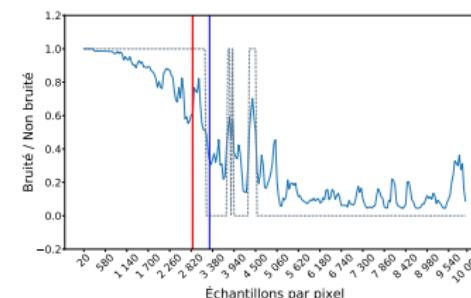
- Score AUC ROC sur la base de test : 82.47%
- $S = 8$ ,  $m = 100$ ,  $F = H_{SVD}$ , et  $n = 40$
- *snorm* comme processus de normalisation

# Modèle de perception RNN : Prédictions

Prédictions correctes du modèle RNN basée sur l'entropie SVD :



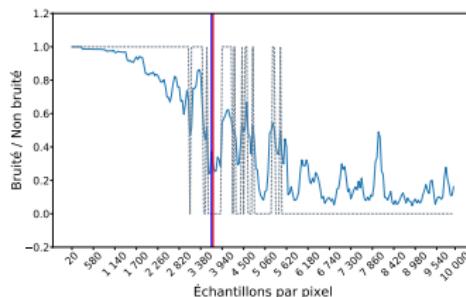
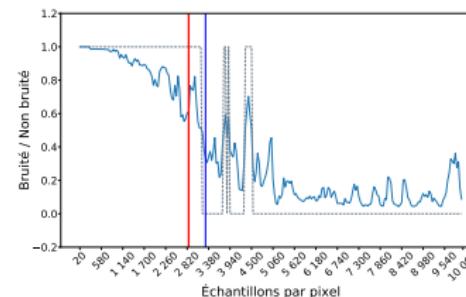
(a) Bloc 5 sur le point de vue *Arc sphere*



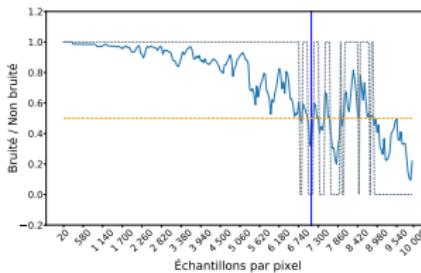
(b) Bloc 8 sur le point de vue *Arc sphere*

# Modèle de perception RNN : Prédictions

## Prédictions correctes du modèle RNN basée sur l'entropie SVD :

(a) Bloc 5 sur le point de vue *Arc sphere*(b) Bloc 8 sur le point de vue *Arc sphere*

## Prédition erronée du modèle RNN basée sur l'entropie SVD :

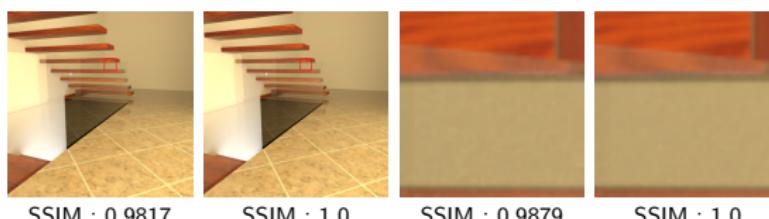
(a) Bloc 9 du point de vue *Classroom*

(b) 7 000 échantillons



(c) 10 000 échantillons

## Modèle de perception RNN : Comparaison visuelle



## Modèle de perception

### Modèle de perception - Generative Adversarial Network (GAN)

## Modèle de perception GGN : Contexte

### Remarques des travaux précédents

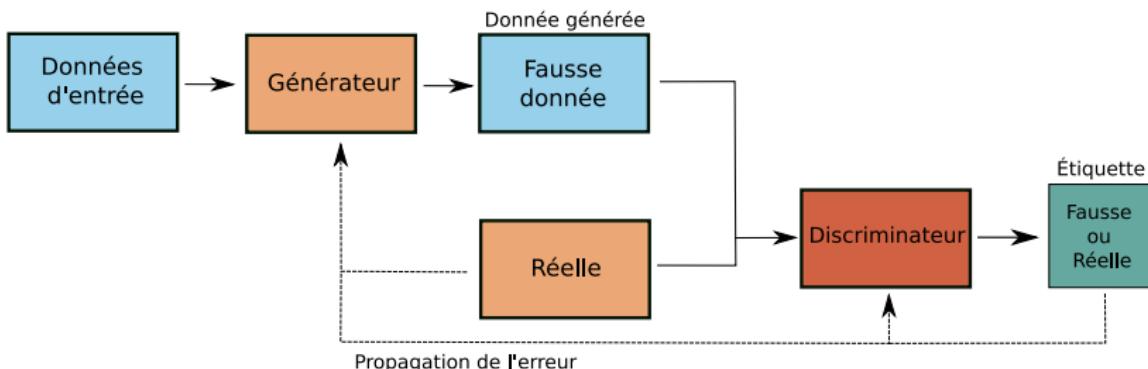
- Image de référence inconnue ou approximée
- Caractérisation du bruit dans une scène difficilement généralisable
- Fenêtre glissante

### Solution envisagée

- Utilisation des méthodes de débruitage
- Génération automatisée des caractéristiques

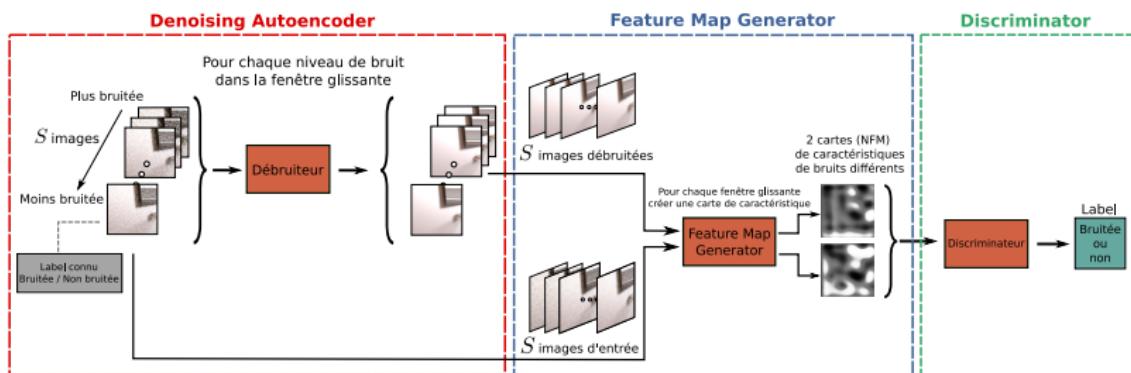
## Modèle de perception GGN : Architecture commune du GAN

**Generative Adversarial Network (GAN)** [Goodfellow et al., 2014] :



# Modèle de perception GGN : Architecture

## Guided-Generative Network (GGN)

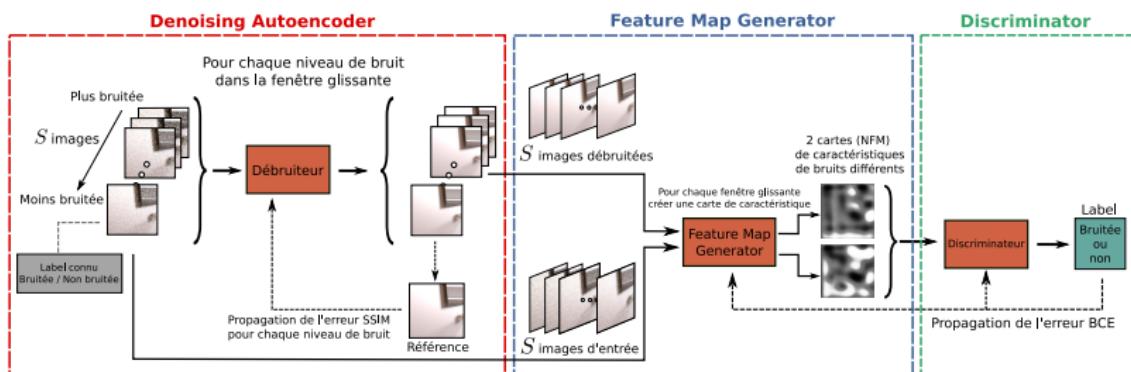


## Objectifs du modèle

- Débruiter correctement la fenêtre d'images [Ronneberger et al., 2015]
- Générer une carte de caractéristiques des différences entre les images de la fenêtre
- Interpréter la différence des cartes de caractéristique

# Modèle de perception GGN : Architecture

## Guided-Generative Network (GGN)



### Comment apprendre un tel modèle ?

- Le modèle de débruitage apprend des images de référence
- Les deux autres modèles apprennent des erreurs du discriminateur
- Utilisation de la fonction de perte *Binary Cross-Entropy* [Buja et al., 2005]

## Modèle de perception GGN : Protocole d'apprentissage

### Base d'apprentissage

- 35 images sélectionnées
- 16 blocs de taille  $200 \times 200$
- 500 différents niveaux de bruit
- Total de 280 000 données étiquetées

## Modèle de perception GGN : Protocole d'apprentissage

### Base d'apprentissage

- 35 images sélectionnées
- 16 blocs de taille  $200 \times 200$
- 500 différents niveaux de bruit
- Total de 280 000 données étiquetées

### Paramètres d'apprentissage

- Fenêtre glissante  $S = 6$
- 30 époques

# Modèle de perception GGN : Résultats du modèle

## Modèle de débruitage :

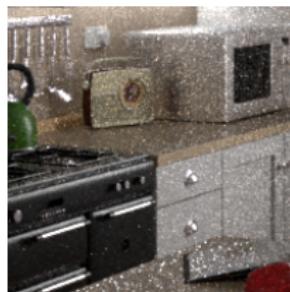


Image bruitée  
SSIM: 0.6433

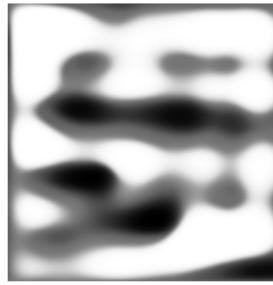


Image débruitée  
SSIM: 0.9859

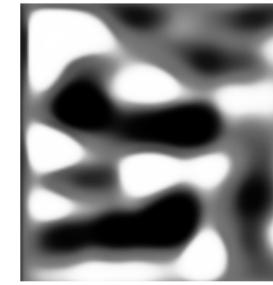


Référence  
SSIM: 1.0

## Modèle de génération de caractéristiques :

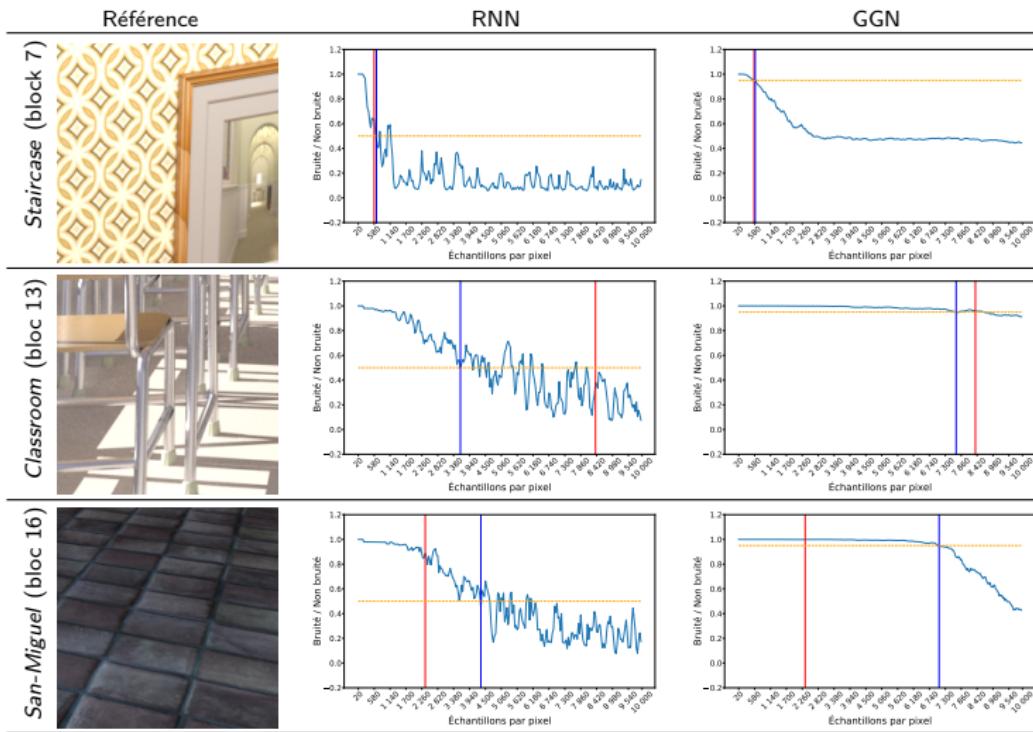


NFM des images bruitées



NFM des références approximées

# Modèle de perception GGN : Prédictions

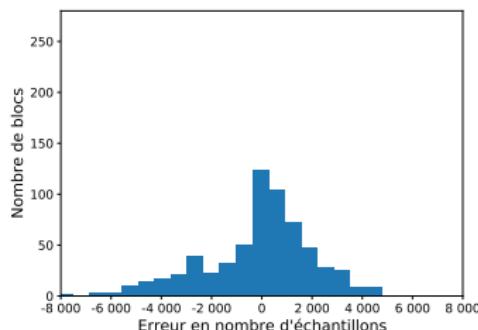


## Modèle de perception

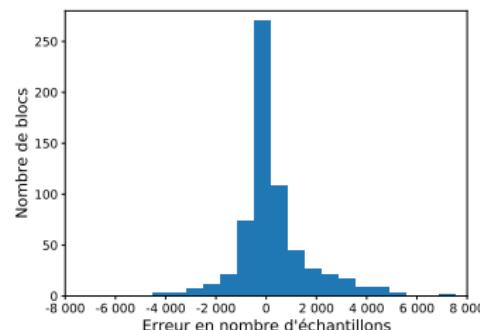
### Modèle de perception - validation des modèles

# Modèle de perception : Étude des performances des modèles

Répartition des écarts entre les seuils prédits et les seuils subjectifs :



(a) RNN - Entropie SVD



(b) GGN

## Modèle de perception : expérience de validation

### Paramètres de l'expérience

- 28 images sélectionnées (aucun seuil à 10 000 échantillons)
- Image reconstruite par rapport aux seuils prédits par le modèle RNN et GGN

# Modèle de perception : expérience de validation

## Paramètres de l'expérience

- 28 images sélectionnées (aucun seuil à 10 000 échantillons)
- Image reconstruite par rapport aux seuils prédits par le modèle RNN et GGN



## Modèle de perception : Résultats

Nombre d'images et pourcentage d'images considérées comme non bruitées :

	RNN - Entropie SVD	GGN
Images	403/476	419/476
Pourcentage	85%	88%

## Modèle de perception

### Modèle RNN

- La SVD entropie caractérise assez bien le bruit MC
- Semble généraliser correctement
- Peut avoir tendance à s'arrêter trop tôt

### Modèle GGN

- Apprend seul à caractériser le bruit
- Possède un léger surapprentissage
- S'arrête généralement après le critère d'arrêt appris

# Modèle de perception

## Modèle RNN

- La SVD entropie caractérise assez bien le bruit MC
- Semble généraliser correctement
- Peut avoir tendance à s'arrêter trop tôt

## Modèle GGN

- Apprend seul à caractériser le bruit
- Possède un léger surapprentissage
- S'arrête généralement après le critère d'arrêt appris

## Remarques globales

- Les deux modèles proposent de bons résultats
- Validation auprès des utilisateurs à hauteur de  $\geq 85\%$

## Publications

- Journal Entropy [Buisine et al., 2021a]
- Conférence *International Conference On Machine Learning And Applications* [Buisine et al., 2021e]

# Plan

## 1 Contexte

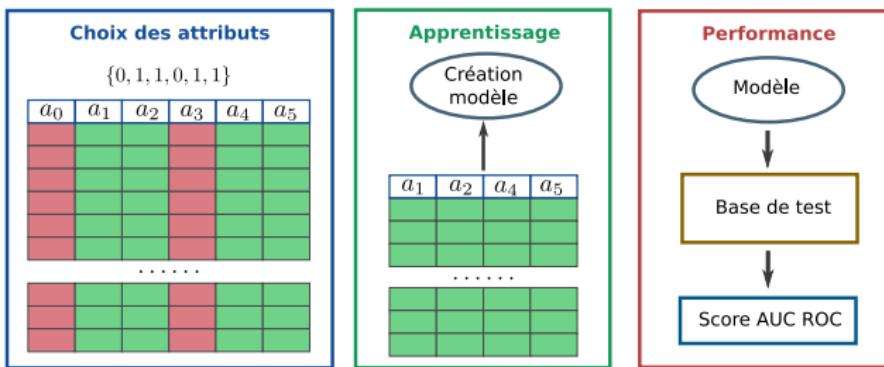
## 2 État de l'art

## 3 Contributions

- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

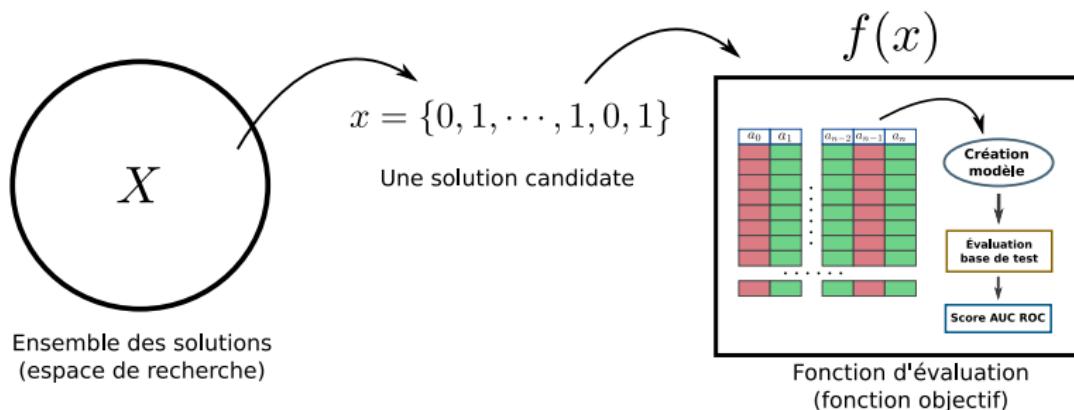
## Sélection d'attributs : Processus classique



### Méthodes existantes

- Importance des variables basées sur les impuretés [Scornet, 2020]
- Importance de la permutation des caractéristiques [Altmann et al., 2010]
- *Recursive Feature Elimination (RFE)* [Yan and Zhang, 2015, Darst et al., 2018]

## Sélection d'attributs : Vers un problème d'optimisation



### Problème d'optimisation pseudo-booléen

$$x^* = \arg \max_{x \in \mathcal{X}} f(x)$$

- $f$ , la fonction objectif (score AUC ROC)
- $x$  est une solution candidate au problème ( $x = (x_1, \dots, x_n)$ )
- $X$  l'espace de recherche
- $x^*$  une solution optimale du problème (meilleure combinaison d'attributs)

## Sélection d'attributs : Optimisation coûteuse

### Problème rencontré : optimisation coûteuse

- $f$  implique un coût en temps conséquent
- Implique une limitation de recherche de solutions

## Sélection d'attributs : Optimisation coûteuse

### Problème rencontré : optimisation coûteuse

- $f$  implique un coût en temps conséquent
- Implique une limitation de recherche de solutions

### Utilisation de métamodèle : Approximer au mieux $f$

- Simulateur de réacteur nucléaire [Muniglia et al., 2017]
- Flux de trafic urbain [Leprêtre et al., 2018]

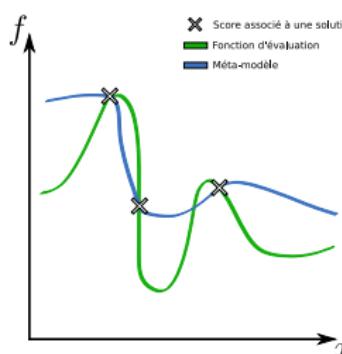
# Sélection d'attributs : Optimisation coûteuse

## Problème rencontré : optimisation coûteuse

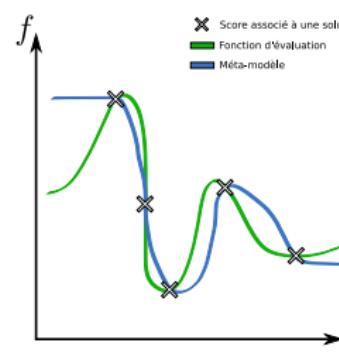
- $f$  implique un coût en temps conséquent
- Implique une limitation de recherche de solutions

## Utilisation de métamodèle : Approximer au mieux $f$

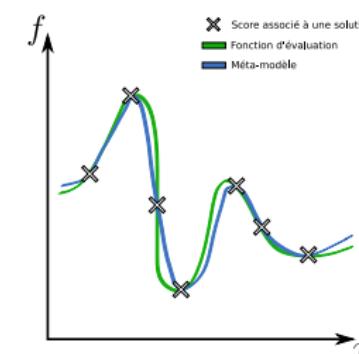
- Simulateur de réacteur nucléaire [Muniglia et al., 2017]
- Flux de trafic urbain [Leprêtre et al., 2018]



(a) Début d'apprentissage



(b) Milieu d'apprentissage



(c) Fin d'apprentissage

## Sélection d'attributs : Méta-modèle à base de Walsh

### Base de Walsh [Walsh, 1923]

$$\varphi_\ell(x) = (-1)^{\sum_{i=1}^n \ell_i x_i}$$

pour tout entier  $\ell$  écrit sous une forme binaire

### Méta-modèle pseudo-booléen [Verel et al., 2018]

$$W_L(x) = \sum_{\ell \text{ s.t. } o(\varphi_\ell) \leq L} a_\ell \varphi_\ell(x)$$

avec  $o$  l'ordre de la fonction de Walsh (le nombre de variables égales à 1 dans la représentation binaire de l'entier  $\ell$ )

## Sélection d'attributs : Méta-modèle à base de Walsh

### Base de Walsh [Walsh, 1923]

$$\varphi_\ell(x) = (-1)^{\sum_{i=1}^n \ell_i x_i}$$

pour tout entier  $\ell$  écrit sous une forme binaire

### Méta-modèle pseudo-booléen [Verel et al., 2018]

$$W_L(x) = \sum_{\substack{\ell \text{ s.t. } o(\varphi_\ell) \leq L}} a_\ell \varphi_\ell(x)$$

avec  $o$  l'ordre de la fonction de Walsh (le nombre de variables égales à 1 dans la représentation binaire de l'entier  $\ell$ )

### Méta-modèle pseudo-booléen à base de Walsh

$$W_2(x) = a_0 + \sum_{i=1}^n a_i (-1)^{x_i} + \sum_{i < j \in N} a_{ij} (-1)^{x_i + x_j}$$

avec  $o = 2$  l'ordre de la fonction de Walsh

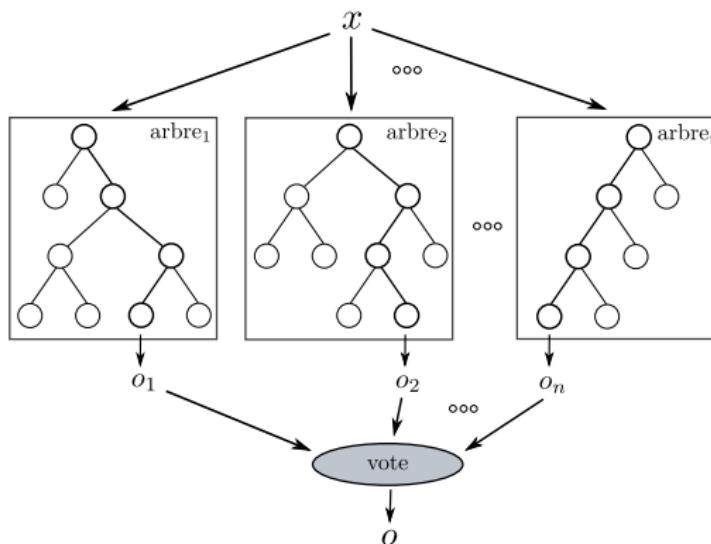
## Sélection d'attributs : Paramètres étudiés

Un total de 32 attributs étudiés

- 26 attributs (écart-type et moyenne) de [Constantin et al., 2015] :
  - filtre linéaire
  - filtre gaussien
  - filtre médian
  - filtre de Wiener adaptatif
  - Décomposition en ondelettes
- 2 attributs de complexité spatielle d'une image [Rousselot et al., 2019]
  - filtre de Sobel
- Complexité de Kolmogorov [Hao et al., 2020]
- L'entropie SVD [Banerjee and Pal, 2014]
- L'écart-type et la moyenne de l'image de luminance  $L$

## Sélection d'attributs : Modèle étudié

### Ensemble d'arbre de décisions

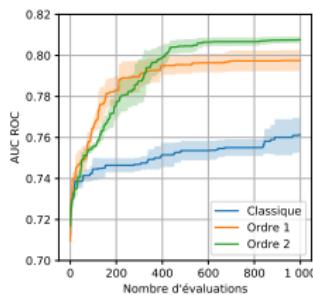
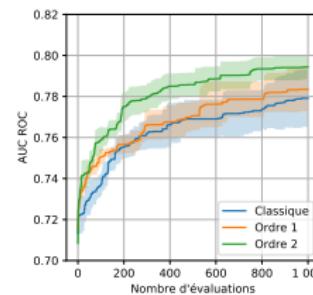
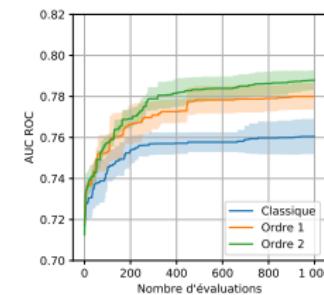


# Sélection d'attributs : Paramètres étudiés et résultats

## Jeux de paramètres

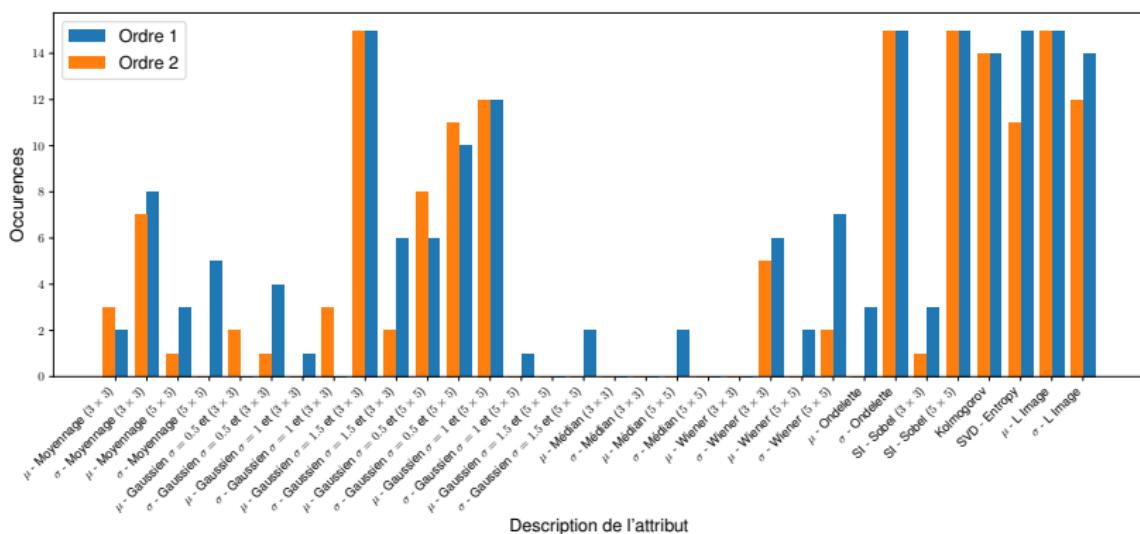
- Le nombre d'évaluations réel maximal à 1 000
- Le nombre d'évaluation par recherche locale avec le méta-modèle,  $LS \in [100, 500, 1000]$
- L'ordre de Walsh  $L \in [1, 2]$

## Performance de l'ordre de Walsh $L$ :

(a)  $LS = 100$ (b)  $LS = 500$ (c)  $LS = 1000$

## Sélection d'attributs : Attributs sélectionnés

## Attributs sélectionnés :



## Sélection d'attributs : Exploitation des résultats

Modèle	<i>Attributs optimisation</i> <sup>1</sup>	<i>Attributs RFE</i>
<b>Forêt</b>	81, 01%	77, 26%
<b>SVM</b>	57, 17%	58, 41%

Scores AUC ROC obtenus sur la base de test

<sup>1</sup>Meilleur modèle retenu sur les différents *runs* après processus d'optimisation

## Sélection d'attributs : Exploitation des résultats

Modèle	<i>Attributs optimisation</i> <sup>1</sup>	<i>Attributs RFE</i>
<b>Forêt</b>	81, 01%	77, 26%
<b>SVM</b>	57, 17%	58, 41%

Scores AUC ROC obtenus sur la base de test

### Hypothèse du problème

- Modèle *simplifiée* de type forêt aléatoire
- Attributs sélectionnés liés au modèle de forêt aléatoire

<sup>1</sup>Meilleur modèle retenu sur les différents *runs* après processus d'optimisation

## Sélection d'attributs : Exploitation des résultats

Modèle	Attributs optimisation <sup>1</sup>	Attributs RFE
Forêt	81, 01%	77, 26%
SVM	57, 17%	58, 41%

Scores AUC ROC obtenus sur la base de test

### Hypothèse du problème

- Modèle *simplifiée* de type forêt aléatoire
- Attributs sélectionnés liés au modèle de forêt aléatoire

### Adaptation au modèle SVM

- Étudier la faisabilité avec un modèle SVM
- Limiter les attributs sélectionnés pour apprentissage (réduction de la complexité)

<sup>1</sup>Meilleur modèle retenu sur les différents *runs* après processus d'optimisation

## Sélection d'attributs : Exploitation des résultats

Modèle	Attributs optimisation <sup>1</sup>	Attributs RFE
Forêt	81, 01%	77, 26%
SVM	57, 17%	58, 41%

Scores AUC ROC obtenus sur la base de test

### Hypothèse du problème

- Modèle *simplifiée* de type forêt aléatoire
- Attributs sélectionnés liés au modèle de forêt aléatoire

### Adaptation au modèle SVM

- Étudier la faisabilité avec un modèle SVM
- Limiter les attributs sélectionnés pour apprentissage (réduction de la complexité)

### Package Python Macop

*Journal of Open Source Software [Buisine et al., 2021c]*

<sup>1</sup>Meilleur modèle retenu sur les différents *runs* après processus d'optimisation

# Plan

## 1 Contexte

## 2 État de l'art

## 3 Contributions

- Base d'images de synthèse
- Gestion des valeurs aberrantes lors du rendu
- Apprentissage profond pour la perception du bruit
- Sélection d'attributs comme un problème d'optimisation

## 4 Conclusion et perspectives

# Conclusion

## Contributions réalisées

- Base d'images assez diversifiées
- Estimateur pour la réduction du bruit nommé *firefly*
- Plusieurs architectures de modèle de perception d'apprentissage profond
- Étude de la sélection des attributs de bruits

# Perspectives

## Extension de la base d'images

- Augmentation de la diversité des points de vue
- Augmentation du nombre total d'échantillons par pixel
- Bruits de plusieurs estimateurs et intégrateurs
- Vers une base d'images stéréoscopiques (stage de Quentin Huan)

# Perspectives

## Extension de la base d'images

- Augmentation de la diversité des points de vue
- Augmentation du nombre total d'échantillons par pixel
- Bruits de plusieurs estimateurs et intégrateurs
- Vers une base d'images stéréoscopiques (stage de Quentin Huan)

## Gestion des valeurs aberrantes

- Étude et comparaisons des autres estimateurs basés MoN  
[Lugosi and Mendelson, 2019, Orenstein, 2019]
- Utilisation des informations non locales
- Extension d'utilisation du Coefficient de Gini
- Étude de la stabilité temporelle de l'estimateur

# Perspectives

## Analyses supplémentaires des modèles de perception

- Étude d'autres méthodes de réduction d'information : ACP, ACI, NMF
- Méthode de réduction du surapprentissage pour le GGN  
[Yazici et al., 2020, Mukherjee et al., 2020]
- Extension des modèles pour la stéréoscopie
- Performances des modèles sur une nature de bruit différente (estimateurs / intégrateurs)

# Perspectives

## Analyses supplémentaires des modèles de perception

- Étude d'autres méthodes de réduction d'information : ACP, ACI, NMF
- Méthode de réduction du surapprentissage pour le GGN  
[Yazici et al., 2020, Mukherjee et al., 2020]
- Extension des modèles pour la stéréoscopie
- Performances des modèles sur une nature de bruit différente (estimateurs / intégrateurs)

## Sélection d'attributs comme un problème d'optimisation

- Étude sur un modèle d'apprentissage plus complexe (SVM)
- Généralisation de la méthode pour un grand nombre d'attributs (bagging)

## Conclusion et perspectives

Merci pour votre attention !

-  Altmann, A., Tološi, L., Sander, O., and Lengauer, T. (2010).  
 Permutation importance: a corrected feature importance measure.  
*26(10):1340–1347.*
-  Bakó, S., Vogels, T., Mcwilliams, B., Meyer, M., Novák, J., Harvill, A.,  
Sen, P., Derose, T., and Rousselle, F. (2017).  
Kernel-predicting convolutional networks for denoising monte carlo  
renderings.  
*36(4):1–14.*
-  Banerjee, M. and Pal, N. R. (2014).  
Feature selection with svd entropy: Some modification and extension.  
*Information Sciences, 264:118–134.*
-  Blair, C. (1985).  
Problem Complexity and Method Efficiency in Optimization (A. S.  
Nemirovsky and D. B. Yudin).  
*SIAM Review, 27(2):264–265.*

-  Bosse, S., Maniry, D., Müller, K.-R., Wiegand, T., and Samek, W. (2017). Deep neural networks for no-reference and full-reference image quality assessment.  
*IEEE Transactions on image processing*, 27(1):206–219.
-  Buisine, J., Bigand, A., Synave, R., Delepoule, S., and Renaud, C. (2021a). Stopping criterion during rendering of computer-generated images based on svd-entropy.  
*Entropy*, 23(1):75.
-  Buisine, J., Delepoule, S., and Renaud, C. (2021b). Firefly Removal in Monte Carlo Rendering with Adaptive Median of meaNs.  
In Bousseau, A. and McGuire, M., editors, *Eurographics Symposium on Rendering - DL-only Track*. The Eurographics Association.
-  Buisine, J., Delepoule, S., and Renaud, C. (2021c). Minimalist and customisable optimisation package.  
*Journal of Open Source Software*, 6(59):2812.

-  Buisine, J., Delepoule, S., Synave, R., and Renaud, C. (2021d).  
Subjective human thresholds over computer generated images.
-  Buisine, J., Teytaud, F., Delepoule, S., and Renaud, C. (2021e).  
Guided-generative network for noise detection in monte-carlo rendering.  
*In International Conference On Machine Learning And Applications.*
-  Buja, A., Stuetzle, W., and Shen, Y. (2005).  
Loss functions for binary class probability estimation and classification:  
Structure and applications.  
*Working draft, November, 3.*
-  Carnec, M., Le Callet, P., and Barba, D. (2008).  
Objective quality assessment of color images based on a generic perceptual  
reduced reference.  
*Signal Processing: Image Communication, 23(4):239–256.*
-  Chaitanya, C. R. A., Kaplanyan, A. S., Schied, C., Salvi, M., Lefohn, A.,  
Nowrouzezahrai, D., and Aila, T. (2017).  
Interactive reconstruction of monte carlo image sequences using a  
recurrent denoising autoencoder.  
*36(4):1–12.*

-  Constantin, J., Bigand, A., Constantin, I., and Hamad, D. (2015).  
Image noise detection in global illumination methods based on FRVM.  
164:82–95.
-  Constantin, J., Constantin, I., Bigand, A., and Hamad, D. (2016).  
Perception of noise in global illumination based on inductive learning.  
In *2016 International Joint Conference on Neural Networks (IJCNN)*,  
pages 5021–5028. IEEE.
-  Damgaard, C. (2003).  
"gini coefficient." from mathworld—a wolfram web resource.
-  Damgaard, C. and Weiner, J. (2000).  
Describing inequality in plant size or fecundity.  
81(4):1139–1142.
-  Darst, B. F., Malecki, K. C., and Engelman, C. D. (2018).  
Using recursive feature elimination in random forest to account for  
correlated variables in high dimensional data.  
19(1):65.

-  DeCoro, C., Weyrich, T., and Rusinkiewicz, S. (2010).  
Density-based outlier rejection in monte carlo rendering.  
*29(7):2119–2125.*
-  Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014).  
Generative adversarial networks.
-  Hampel, F. R. (1971).  
A general qualitative definition of robustness.  
*The Annals of Mathematical Statistics, 42(6):1887–1896.*
-  Hao, Q., Ma, L., Sbert, M., Feixas, M., and Zhang, J. (2020).  
Gaze information channel in van gogh's paintings.  
*Entropy, 22(5):540.*
-  Hinton, G. E. and Salakhutdinov, R. R. (2006).  
Reducing the dimensionality of data with neural networks.  
*313(5786):504–507.*
-  Hochreiter, S. and Schmidhuber, J. (1997).  
Long short-term memory.  
*Neural computation, 9(8):1735–1780.*

-  **Huo, Y. and Yoon, S.-e. (2021).**  
A survey on deep learning-based monte carlo denoising.  
*7(2):169–185.*
-  **Jerrum, M. R., Valiant, L. G., and Vazirani, V. V. (1986).**  
Random generation of combinatorial structures from a uniform distribution.  
*Theoretical Computer Science*, 43:169–188.
-  **Jung, J. W., Meyer, G., and DeLong, R. (2015).**  
Robust statistical pixel estimation.  
*34(2):585–596.*
-  **Kajiya, J. T. (1986).**  
The rendering equation.  
In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150.
-  **Kang, L., Ye, P., Li, Y., and Doermann, D. (2014).**  
Convolutional neural networks for no-reference image quality assessment.  
In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1733–1740. IEEE.

-  Leprêtre, F., Verel, S., Fonlupt, C., and Marion, V. (2019).  
Walsh functions as surrogate model for pseudo-boolean optimization problems.  
In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 303–311.
-  Leprêtre, F., Fonlupt, C., Verel, S., and Marion, V. (2018).  
SIALAC benchmark: on the design of adaptive algorithms for traffic lights problems.  
In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '18, pages 288–289. Association for Computing Machinery.
-  Li, T.-M., Wu, Y.-T., and Chuang, Y.-Y. (2012).  
SURE-based optimization for adaptive sampling and reconstruction.  
31(6):1.
-  Liu, J., Pérez-Liébana, D., and Lucas, S. M. (2017).  
Bandit-based random mutation hill-climbing.  
In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 2145–2151. IEEE.

-  Liu, W. (2014).  
Additive white gaussian noise level estimation based on block svd.  
In *2014 IEEE Workshop on Electronics, Computer and Applications*, pages 960–963. IEEE.
-  Lugosi, G. and Mendelson, S. (2019).  
Risk minimization by median-of-means tournaments.
-  Mantiuk, R., Kim, K. J., Rempel, A. G., and Heidrich, W. (2011).  
HDR-VDP-2: a calibrated visual metric for visibility and quality predictions in all luminance conditions.  
*30(4):40:1–40:14.*
-  Mikolov, T., Karafiat, M., Burget, L., Cernocky, J., and Khudanpur, S. (2010).  
Recurrent neural network based language model.  
In *Interspeech*, volume 2, pages 1045–1048. Makuhari.
-  Moorthy, A. K. and Bovik, A. C. (2011).  
Blind image quality assessment: From natural scene statistics to perceptual quality.  
*20(12):3350–3364.*

-  Mukherjee, S., Xu, Y., Trivedi, A., and Ferres, J. L. (2020).  
Protecting gans against privacy attacks by preventing overfitting.
-  Muniglia, M., Verel, S., Pallec, J.-C. L., and Do, J.-M. (2017).  
Massive asynchronous master-worker ea for nuclear reactor optimization: a fitness landscape perspective.  
In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 295–296.
-  Myszkowski, K. (1998).  
The visible differences predictor: applications to global illumination problems.  
In Drettakis, G. and Max, N., editors, *Rendering Techniques '98*, pages 223–236. Springer Vienna.
-  Narwaria, M., Mantiuk, R. K., Da Silva, M. P., and Le Callet, P. (2015).  
HDR-VDP-2.2: a calibrated method for objective quality prediction of high-dynamic range and standard images.  
*24(1):010501.*

-  Orenstein, P. (2019).  
Robust Mean Estimation with the Bayesian Median of Means.  
*arXiv:1906.01204 [math, stat]*.  
arXiv: 1906.01204.
-  Pharr, M., Jakob, W., and Humphreys, G. (2016).  
*Physically Based Rendering: From Theory to Implementation*.  
Morgan Kaufmann.  
Google-Books-ID: iNMVBQAAQBAJ.
-  Ponomarenko, N., Jin, L., Ieremeiev, O., Lukin, V., Egiazarian, K., Astola, J., Vozel, B., Chehdi, K., Carli, M., Battisti, F., and Jay Kuo, C.-C. (2015).  
Image database TID2013: Peculiarities, results and perspectives.  
30:57–77.
-  Ronneberger, O., Fischer, P., and Brox, T. (2015).  
U-net: Convolutional networks for biomedical image segmentation.  
In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, pages 234–241. Springer International Publishing.

- Rousselle, F., Manzi, M., and Zwicker, M. (2013).  
Robust denoising using feature and color information.  
*32(7):121–130.*
- Rousselot, M., Le Meur, O., Cozot, R., and Ducloux, X. (2019).  
Quality assessment of hdr/wcg images using hdr uniform color spaces.  
*Journal of Imaging*, 5(1):18.
- Scornet, E. (2020).  
Trees, forests, and impurity-based variable importance.
- Sharma, S., Sharma, S., and Athaiya, A. (2017).  
Activation functions in neural networks.  
*towards data science*, 6(12):310–316.
- Sheikh, H. R., Bovik, A. C., and Cormack, L. (2005).  
No-reference quality assessment using natural scene statistics: Jpeg2000.  
*IEEE Transactions on image processing*, 14(11):1918–1927.

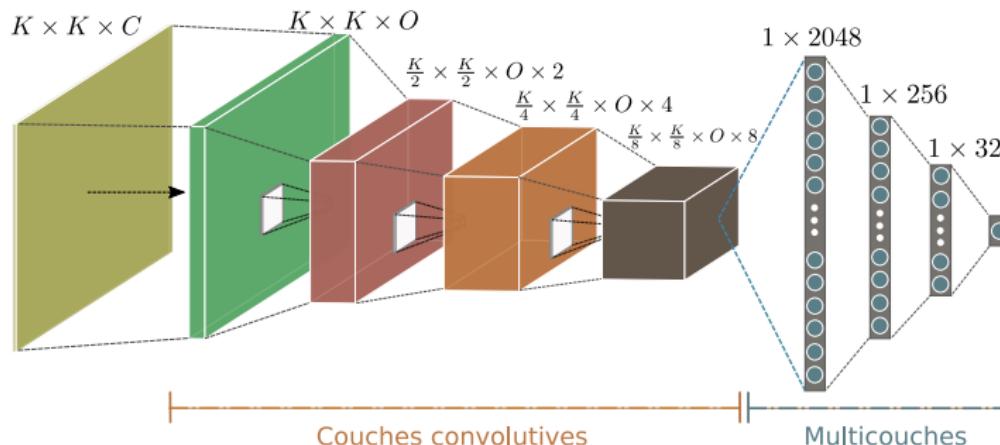
-  Takouachet, N., Delepoulle, S., Renaud, C., Zoghlami, N., and Tavares, J. M. R. (2017). Perception of noise and global illumination: Toward an automatic stopping criterion based on SVM. 69:49–58.
-  Verel, S., Derbel, B., Liefoghe, A., Aguirre, H., and Tanaka, K. (2018). A surrogate model based on walsh decomposition for pseudo-boolean functions. In *International conference on parallel problem solving from nature*, pages 181–193. Springer.
-  Vogels, T., Rousselle, F., Mcwilliams, B., Röthlin, G., Harvill, A., Adler, D., Meyer, M., and Novák, J. (2018). Denoising with kernel prediction and asymmetric loss functions. 37(4):1–15.
-  Walsh, J. L. (1923). A closed set of normal orthogonal functions. *American Journal of Mathematics*, 45(1):5–24.

-  Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004).  
Image quality assessment: From error visibility to structural similarity.  
13(4):600–612.
-  Welstead, S. T. (1999).  
*Fractal and wavelet image compression techniques*, volume 40.  
Spie Press.
-  Xu, B., Zhang, J., Wang, R., Xu, K., Yang, Y.-L., Li, C., and Tang, R. (2019).  
Adversarial monte carlo denoising with conditioned auxiliary feature modulation.  
38(6):224:1–224:12.
-  Yan, K. and Zhang, D. (2015).  
Feature selection and analysis on correlated gas sensor data with recursive feature elimination.  
212:353–363.

-  Yazici, Y., Foo, C.-S., Winkler, S., Yap, K.-H., and Chandrasekhar, V. (2020).  
Empirical analysis of overfitting and mode drop in gan training.  
In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1651–1655. IEEE.
-  Ye, P., Kumar, J., Kang, L., and Doermann, D. (2013).  
Real-time no-reference image quality assessment based on filter learning.  
In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 987–994.  
ISSN: 1063-6919.
-  Zirr, T., Hanika, J., and Dachsbaecher, C. (2018).  
Re-weighting firefly samples for improved finite-sample monte carlo estimates.  
*37*(6):410–421.

# État de l'art : Les différents réseaux de neurones

## Réseau de neurones convolutionnel (CNN<sup>1</sup>) :



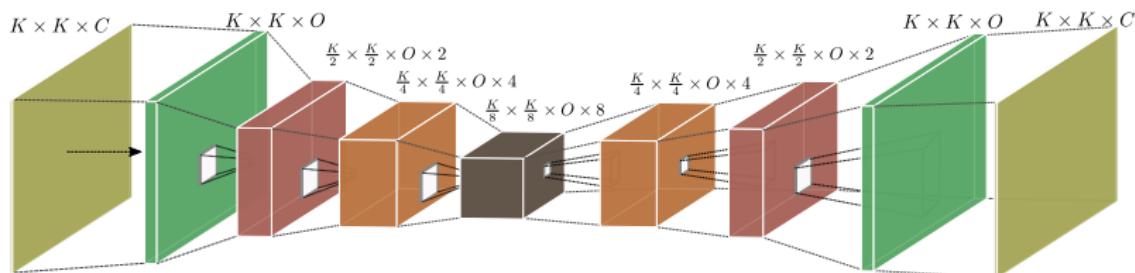
## Applications de classification

- Reconnaissance de caractère
- Classification d'espèces

<sup>1</sup>Convolutional Neural Network

# État de l'art : Les différents réseaux de neurones

Réseau de neurones Autoencoder [Hinton and Salakhutdinov, 2006] :

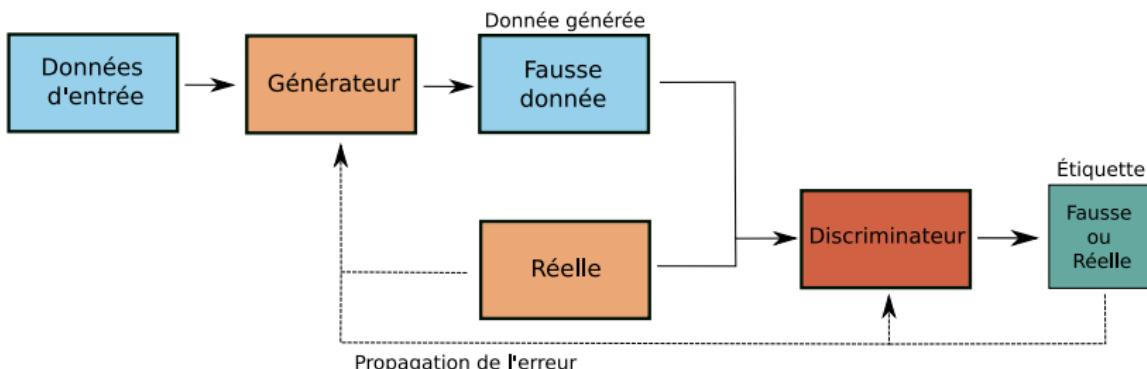


## Applications

- Débruitage d'images
- Compression d'image
- Extraction de caractéristiques
- ...

## État de l'art : Les différents réseaux de neurones

**Generative Adversarial Network (GAN) [Goodfellow et al., 2014] :**

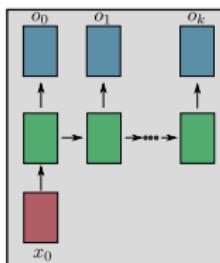


### Applications

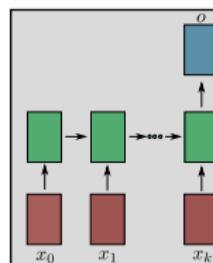
- Génération d'images
- Transformation d'images
- ...

# État de l'art : Les différents réseaux de neurones

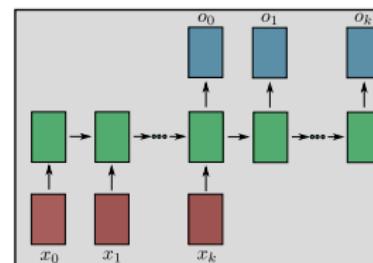
## Réseaux de neurones récurrents [Mikolov et al., 2010] :



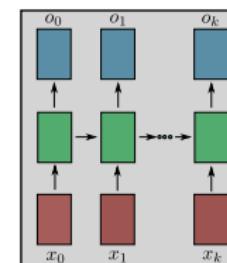
(a) Prédictions de mots



(b) Classification /  
Prédiction de mot



(c) Prédictions de mots



(d) Traduction de  
texte

# État de l'art : Applications de débruitage pour le rendu

## Processus de post-traitement :



## État de l'art : Applications de débruitage pour le rendu

### Processus de post-traitement :



### Différentes architectures [Huo and Yoon, 2021]

- Réseaux convolutifs [Li et al., 2012, Rousselle et al., 2013, Bako et al., 2017]
- Autoencoder récurrent pour séquence d'images [Chaitanya et al., 2017, Vogels et al., 2018]
- GAN pour le débruitage [Xu et al., 2019]

## État de l'art : Applications de débruitage pour le rendu

### Processus de post-traitement :



### Différentes architectures [Huo and Yoon, 2021]

- Réseaux convolutifs [Li et al., 2012, Rousselle et al., 2013, Bako et al., 2017]
- Autoencoder récurrent pour séquence d'images [Chaitanya et al., 2017, Vogels et al., 2018]
- GAN pour le débruitage [Xu et al., 2019]

### Avantages

- Débruitage rapide de l'image
- Nécessite peu d'échantillons par pixel
- Utilise les informations de la scène

# État de l'art : Applications de débruitage pour le rendu

## Processus de post-traitement :



Différentes architectures [Huo and Yoon, 2021]

- Réseaux convolutifs [Li et al., 2012, Rousselle et al., 2013, Bako et al., 2017]
- Autoencoder récurrent pour séquence d'images [Chaitanya et al., 2017, Vogels et al., 2018]
- GAN pour le débruitage [Xu et al., 2019]

### Avantages

- Débruitage rapide de l'image
- Nécessite peu d'échantillons par pixel
- Utilise les informations de la scène

### Inconvénients

- Nécessite une grande quantité de données
- Peut amener à une perte de certains effets lumineux
- Ajoute un biais à l'image finale obtenue

## Backup : métriques

### Type de tâche

- Classification
- Régression

## Backup : métriques

### Type de tâche

- Classification
- Régression

$$y' = h(x)$$

avec  $y' \in \{0, 1\}$

$$\text{Justesse} = \frac{VP + VN}{VP + VN + FP + FN}$$

$$\text{Précision} = \frac{VP}{VP + FP} \quad \text{Rappel} = \frac{VP}{VP + FN}$$

$$TVP = \frac{VP}{VP + FN} \quad TFP = \frac{FP}{FP + VN}$$

Avec  $VP = \text{Vrais positifs}$ ,  $VN = \text{Vrais négatifs}$ ,  $FP = \text{Faux positifs}$ ,  
et  $FN = \text{Faux négatifs}$ .

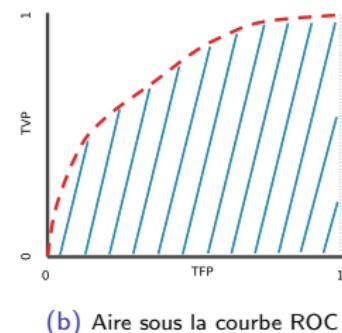
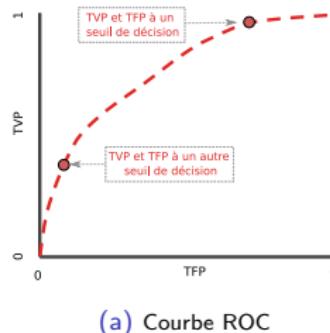
# Backup : métriques

## Type de tâche

- Classification
- Régression

$$y' = h(x)$$

avec  $y' \in \{0, 1\}$



## Backup : métriques

### Type de tâche

- Classification
- Régression

$$y' = h(x)$$

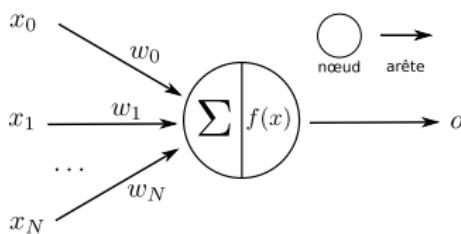
avec  $y' \in \mathbb{R}$

$$\text{EMA} = \frac{\sum_{i=1}^n |y_i - y'_i|}{n}$$

$$\text{EQM} = \frac{\sum_{i=1}^n (y_i - y'_i)^2}{n}$$

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - y'_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

## Backup : Réseau de neurone

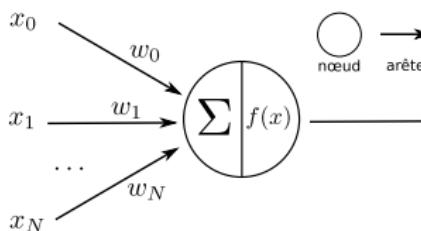
**Modèle perceptron :**

---

¹Support Vector Machine

## Backup : Réseau de neurone

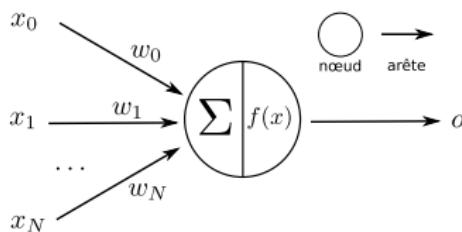
### Modèle perceptron :



$$o = f(x) = \begin{cases} 1 & \text{si} \\ 0 & \text{sinon} \end{cases} \quad \sum_{i=1}^n w_i x_i > \theta$$

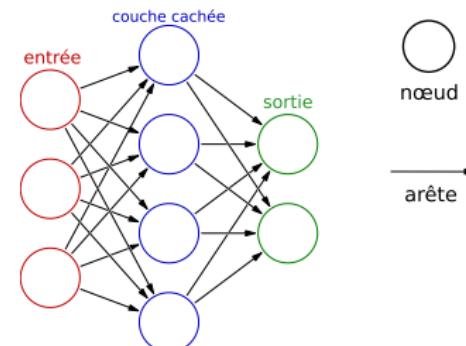
## Backup : Réseau de neurone

## Modèle perceptron :



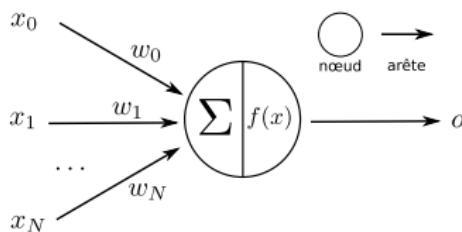
$$o = f(x) = \begin{cases} 1 & \text{si} \\ 0 & \text{sinon} \end{cases} \quad \sum_{i=1}^n w_i x_i > \theta$$

## Réseau de neurones :



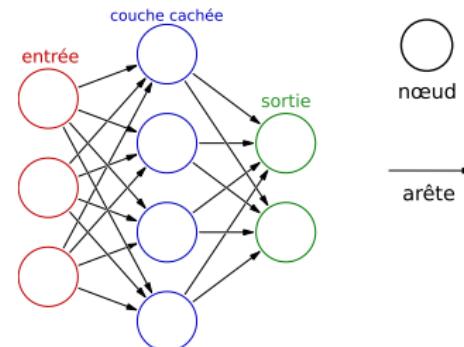
## Backup : Réseau de neurone

## Modèle perceptron :



$$o = f(x) = \begin{cases} 1 & \text{si } \sum_{i=1}^n w_i x_i > \theta \\ 0 & \text{sinon} \end{cases}$$

## Réseau de neurones :



## Fonction de perte

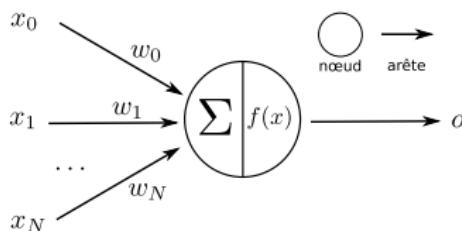
$$w_i = w_i + \alpha F(y, f(x))x_i$$

---

<sup>1</sup>Support Vector Machine

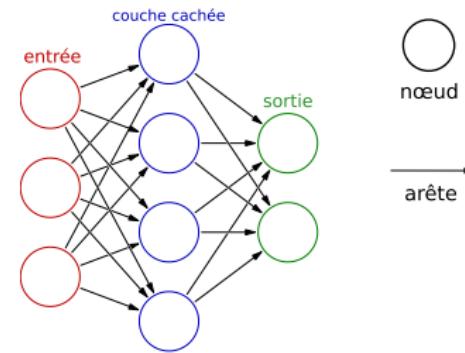
## Backup : Réseau de neurone

## Modèle perceptron :



$$o = f(x) = \begin{cases} 1 & \text{si} \\ 0 & \text{sinon} \end{cases} \quad \sum_{i=1}^n w_i x_i > \theta$$

## Réseau de neurones :



## Fonction de perte

$$w_i = w_i + \alpha F(y, f(x))x_i$$

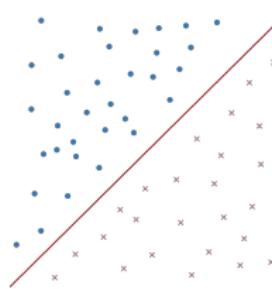
## Fonction d'activation [Sharma et al., 2017]

- Sigmoïde
- Tangente hyperbolique ( $\tanh$ )
- *Rectified Linear Unit* (ReLU)
- Softmax

---

<sup>1</sup>Support Vector Machine

## Backup : modèle SVM

**Modèle de type machine à vecteur support (SVM<sup>1</sup>)**

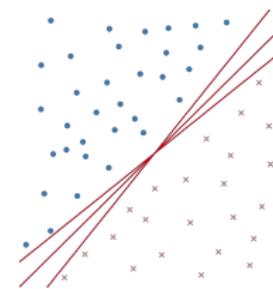
(a) Séparateur linéaire

**Séparateur linéaire :**

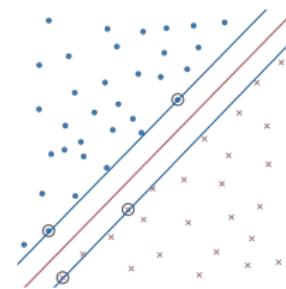
$$h(x) = w^T x + w_0$$

**Avec :**

- vecteur d'entrée  $x = (x_1, \dots, x_N)^T$
- vecteur de poids  $w = (w_1, \dots, w_N)^T$



(b) Possibilité de séparations



(c) Marge maximale

**Kernel trick :**  $K(x_i, x_j) = \phi(x_i)^T \cdot \phi(x_j)$ 

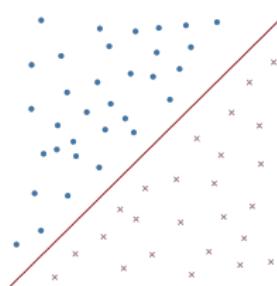
- Polynomial :  $K(x_i, x_j) = (x_i^T \cdot x_j + 1)^d$
- Gaussien :  $K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$
- Radial basis function :  $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$

**Marge maximale :**

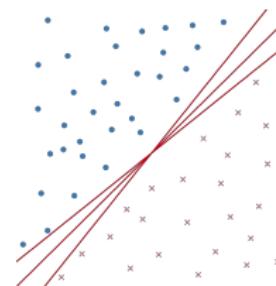
$$\arg \max_{w, w_0} \min_k \left\{ \|x - x_k\| : x \in \mathbb{R}^N, w^T x + w_0 = 0 \right\}$$

<sup>1</sup>Support Vector Machine

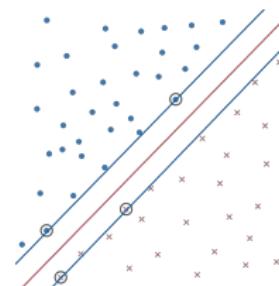
## Backup : modèle SVM

**Modèle de type machine à vecteur support (SVM<sup>1</sup>)**

(a) Séparateur linéaire



(b) Possibilité de séparations



(c) Marge maximale

**Séparateur linéaire :**

$$h(x) = w^T x + w_0$$

Avec :

- vecteur d'entrée  $x = (x_1, \dots, x_N)^T$
- vecteur de poids  $w = (w_1, \dots, w_N)^T$

**Marge maximale :**

$$\arg \max_{w, w_0} \min_k \left\{ \|x - x_k\| : x \in \mathbb{R}^N, w^T x + w_0 = 0 \right\}$$

**Cas linéairement non séparable :**

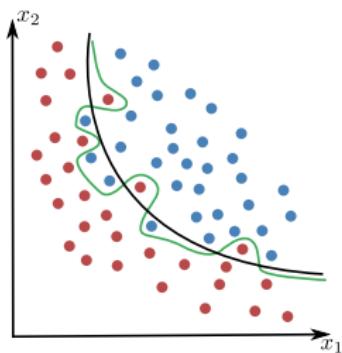
- Kernel de redescription
- $C$  le paramètre de la marge souple
- $\gamma$  paramètre l'importance des données dans l'espace de redescription

---

<sup>1</sup>Support Vector Machine

# Apprentissage automatique : surapprentissage

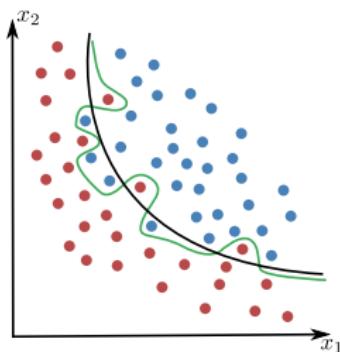
## Problème du surapprentissage d'un modèle



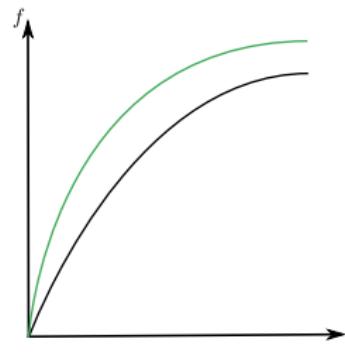
(a) Performance classification

## Apprentissage automatique : surapprentissage

### Problème du surapprentissage d'un modèle



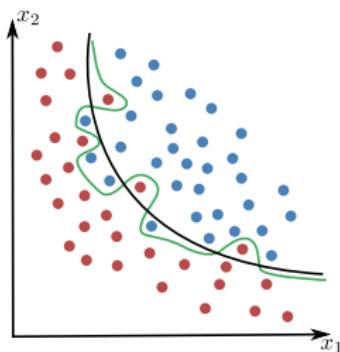
(a) Performance classification



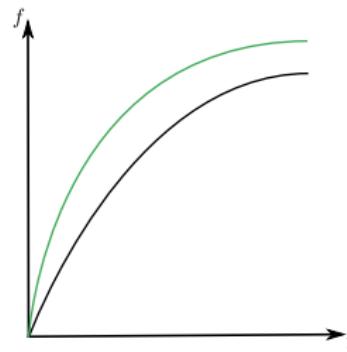
(b) Performance d'apprentissage

# Apprentissage automatique : surapprentissage

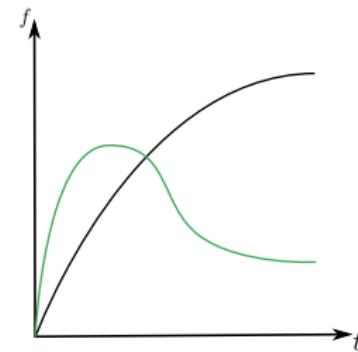
## Problème du surapprentissage d'un modèle



(a) Performance classification



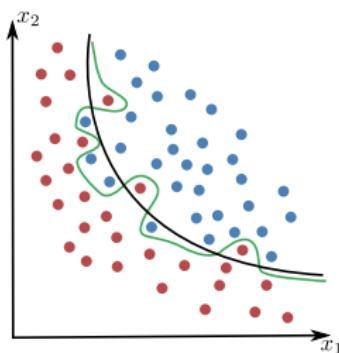
(b) Performance d'apprentissage



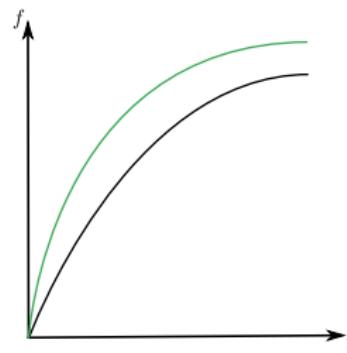
(c) Performance de validation

## Apprentissage automatique : surapprentissage

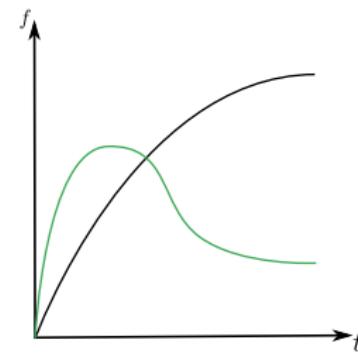
### Problème du surapprentissage d'un modèle



(a) Performance classification



(b) Performance d'apprentissage



(c) Performance de validation

### Palier le surapprentissage

- **Validation croisée** :  $k$  sous-ensembles où  $k - 1$  ensembles sont utilisées pour l'apprentissage
- **Dropout** : pourcentage de neurones d'une couche du réseau où l'erreur n'est pas propagée

## Backup : Mesures de qualité d'images

### Bases de données d'images naturelles

- LIVE [Sheikh et al., 2005]
- TID [Ponomarenko et al., 2015]

### Contenu

- Images naturelles
- Bruits additifs
- Score MOS (Mean Opinion Score)

## Backup : Mesures de qualité d'images

### Bases de données d'images naturelles

- LIVE [Sheikh et al., 2005]
- TID [Ponomarenko et al., 2015]

### Contenu

- Images naturelles
- Bruits additifs
- Score MOS (Mean Opinion Score)

---

### Classes de mesures

- Avec référence
- Référence réduite
- Sans référence

## Backup : Mesures de qualité d'images

### Bases de données d'images naturelles

- LIVE [Sheikh et al., 2005]
- TID [Ponomarenko et al., 2015]

### Contenu

- Images naturelles
- Bruits additifs
- Score MOS (Mean Opinion Score)

### Classes de mesures

- Avec référence
- Référence réduite
- Sans référence

- $PSNR = 10 \cdot \log_{10} \left( \frac{d^2}{EQM} \right)$  [Welstead, 1999]
- *Structural Similarity Index Measure (SSIM)* [Wang et al., 2004]
- *HDR-VDP (Visual Discrimination Model)* [Mantiuk et al., 2011, Narwaria et al., 2015]

## Backup : Mesures de qualité d'images

### Bases de données d'images naturelles

- LIVE [Sheikh et al., 2005]
- TID [Ponomarenko et al., 2015]

### Contenu

- Images naturelles
- Bruits additifs
- Score MOS (Mean Opinion Score)

### Classes de mesures

- Avec référence
- Référence réduite
- Sans référence

- DIIVINE<sup>1</sup> [Moorthy and Bovik, 2011]
- Basé modèle d'apprentissage SVR<sup>2</sup> [Ye et al., 2013]
- Basé modèle d'apprentissage profond [Kang et al., 2014, Bosse et al., 2017]

<sup>1</sup>Distortion Identification-based Image Verity and INtegrity Evaluation

<sup>2</sup>Support Vector Regression

## Backup : Mesures de qualité d'images

Comparaisons de SSIM et du HDR-VDP sur un point de vue de la scène Bedroom



(a) 100 échantillons  
SSIM: 0.6315



(b) 10 000 échantillons  
SSIM: 1.0



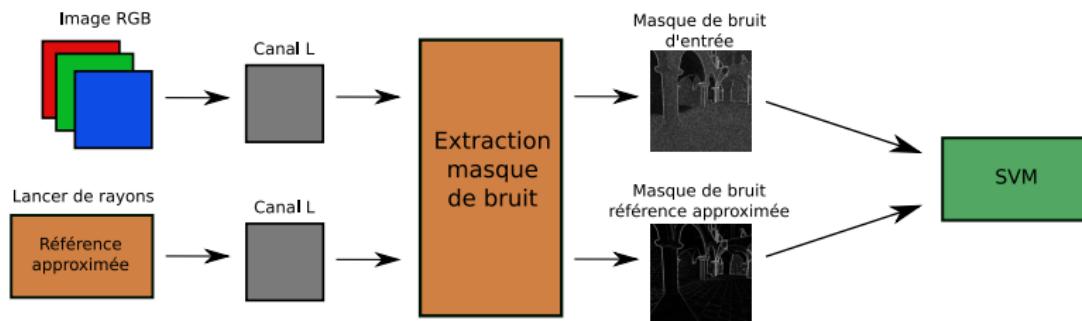
(c) Carte de probabilité  
MOS: 53.814

### Problèmes soulevés

- Métriques axées sur des bruits d'images naturelles
- Nécessite une images de référence pour les plus connues

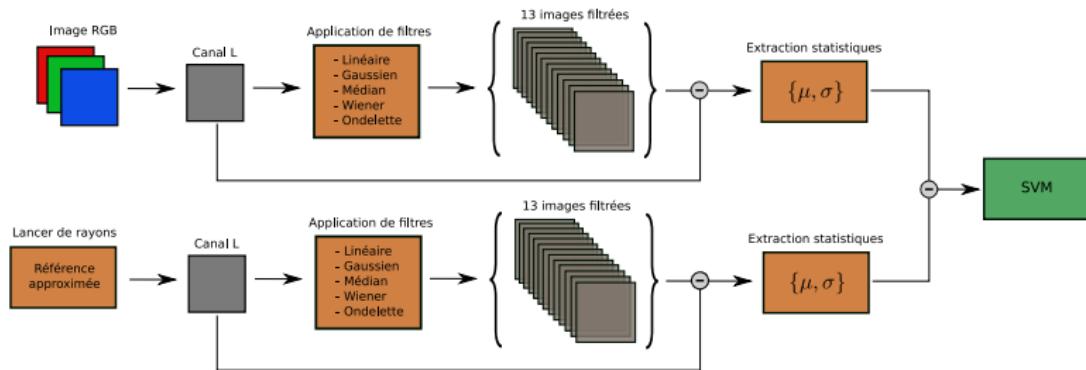
## Backup : Propositions de modèles

## Approche de [Takouachet et al., 2017]



## Backup : Propositions de modèles

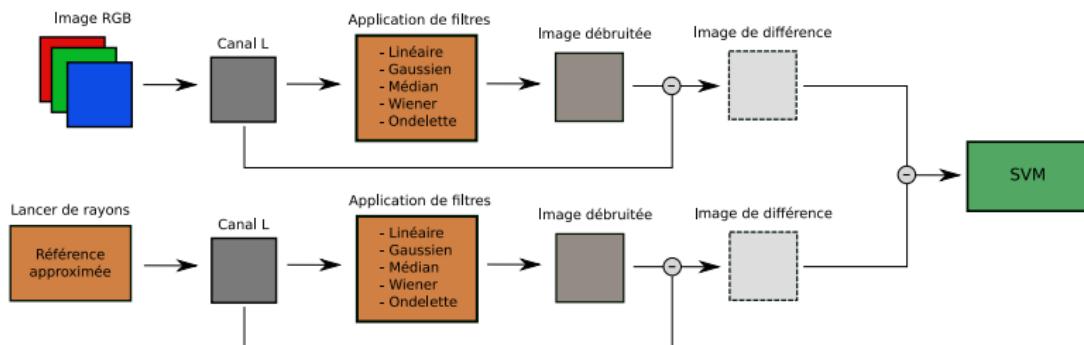
## Caractéristiques proposées par l'approche de [Constantin et al., 2015]



type de filtre	taille de la convolution	
	$3 \times 3$	$5 \times 5$
filtre linéaire	image 1	image 2
filtre gaussien	$\sigma = 0,5$	image 3
	$\sigma = 1,0$	image 4
	$\sigma = 1,5$	image 5
filtre médian	image 9	image 10
filtre de Wiener adaptatif	image 11	image 12
ondelettes	image 13	

## Backup : Propositions de modèles

### Approche proposée par [Constantin et al., 2016]



## Backup : MoN

**L'estimateur MoN** [Blair, 1985, Jerrum et al., 1986] :

$$\hat{\mu}_{MoN} = \text{médiane}\left(\frac{1}{k} \sum_{i=1}^k x_i, \dots, \frac{1}{k} \sum_{i=n-k+1}^n x_i\right)$$

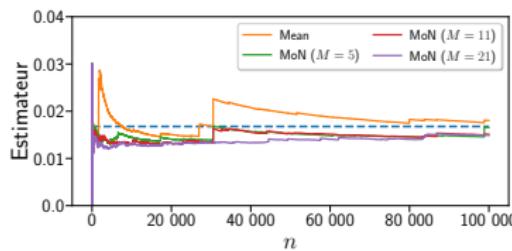
avec des ajustements mineurs si  $\frac{n}{k}$  n'est pas un nombre entier.

## Backup : MoN

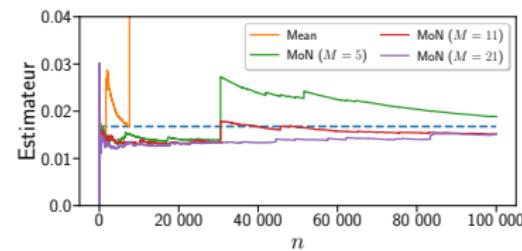
**L'estimateur MoN** [Blair, 1985, Jerrum et al., 1986] :

$$\hat{\mu}_{MoN} = \text{médiane}\left(\frac{1}{k} \sum_{i=1}^k x_i, \dots, \frac{1}{k} \sum_{i=n-k+1}^n x_i\right)$$

avec des ajustements mineurs si  $\frac{n}{k}$  n'est pas un nombre entier.



(a) Sans valeurs aberrantes



(b) Avec valeurs aberrantes

## Valeurs aberrantes : Jung et al.

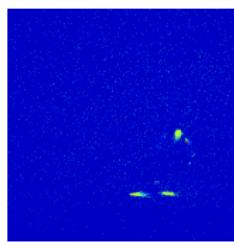
Paramètre automatique du nombre  $M$  de moyennes [Jung et al., 2015] :

$$M = \begin{cases} 1 & \text{pour } \frac{\sigma_{pixel}}{\sigma_{image}} < 1, \\ 2 \times \left( \lfloor \log_2 \frac{\sigma_{pixel}}{\sigma_{image}} \rfloor + 1 \right) + 1 & \text{sinon} \end{cases} \quad (1)$$

## Valeurs aberrantes : Jung et al.

Paramètre automatique du nombre  $M$  de moyennes [Jung et al., 2015] :

$$M = \begin{cases} 1 & \text{pour } \frac{\sigma_{pixel}}{\sigma_{image}} < 1, \\ 2 \times \left( \lfloor \log_2 \frac{\sigma_{pixel}}{\sigma_{image}} \rfloor + 1 \right) + 1 & \text{sinon} \end{cases} \quad (1)$$



(a) Carte de chaleur  
du paramètre  $M$



(b) Jung et al.



(c) Moyenne



(d) MoN ( $M = 17$ )

## Backup : reformulation du coefficient de Gini

Dérivé de la formulation fournie dans  
[Damgaard and Weiner, 2000, Damgaard, 2003] :

$$\begin{aligned} G &= 2 \frac{\sum_{i=1}^n i \cdot x_i}{n^2 \mu} - \frac{(n+1) \sum_{i=1}^n x_i}{n^2 \mu} \\ &= 2 \frac{\sum_{i=1}^n i \cdot x_i}{n^2 \frac{1}{n} \sum_{i=1}^n x_i} - (n+1) \frac{1}{n} \sum_{i=1}^n x_i \times \frac{1}{n \mu} \quad (2) \\ &= \frac{2 \sum_{i=1}^n i \cdot x_i}{n \sum_{i=1}^n x_i} - (n+1) \frac{\mu}{n \mu} \\ &= \frac{2 \sum_{i=1}^n i \cdot x_i}{n \sum_{i=1}^n x_i} - \frac{n+1}{n} \end{aligned}$$

## Modèle de perception : Sources d'erreurs

Problème de pixels identifiés comme firefly :



(a) Image modèle GGN



(b) Bloc modèle GGN



(c) Bloc référence humaine

## Backup : Comparaison modèle de perception RNN

### Problématique de comparaison aux travaux précédents basés sur un SVM

#### Complexité d'apprentissage :

$$O(n_f \times n_o^2)$$

- $n_f$  représente le nombre de caractéristiques d'entrée d'une donnée
- $n_o$  le nombre de données en entrée du modèle

#### Autre problématique :

- Trouver les meilleurs paramètres de l'hyper-plan séparateur
- Utilisation de la validation croisée

### Comparaison avec la méthode proposée par [Constantin et al., 2015]

- Utilisation des 26 attributs
- Pas d'image de référence mais utilisation de la fenêtre glissante

## Backup : Comparaison modèle de perception RNN

### Problématique de comparaison aux travaux précédents basés sur un SVM

#### Complexité d'apprentissage :

$$O(n_f \times n_o^2)$$

- $n_f$  représente le nombre de caractéristiques d'entrée d'une donnée
- $n_o$  le nombre de données en entrée du modèle

#### Autre problématique :

- Trouver les meilleurs paramètres de l'hyper-plan séparateur
- Utilisation de la validation croisée

### Comparaison avec la méthode proposée par [Constantin et al., 2015]

- Utilisation des 26 attributs
- Pas d'image de référence mais utilisation de la fenêtre glissante

### Modèle retenu pour [Constantin et al., 2015]

- Score AUC ROC sur la base de test : 80.78%
- $S = 8$ ,  $snorm$  et  $n = 20$

# Backup : Nouvelles prédictions

11820	11060	9580	7860
11620	7060	9540	6940
8540	6420	6340	8340
5420	6140	6940	8300

(a) RNN - *Living-room*  
SSIM : 0.8933

20380	30760	30120	26460
12120	18000	39880	20360
16400	32920	64000	32000
36440	42620	22960	29940

(b) GGN - *Living-room*  
SSIM : 0.9407

9073	10000	10000	7433
8806	6086	9206	6820
6620	8406	10000	10000
9006	9740	10000	10000

(c) Seuils humains



(d) 64 000 échantillons

11220	14380	14860	14060
7900	14020	11700	12980
7220	8820	9780	9340
6980	6140	7340	8420

(e) RNN - *Classroom*  
SSIM : 0.8973

21640	11980	26920	34640
36800	28460	43940	29900
17420	29400	32920	20360
28920	35320	26240	51920

(f) GGN - *Classroom*  
SSIM : 0.9461

10000	10000	10000	10000
8220	10000	10000	10000
10000	9280	10000	9220
8346	9606	8073	9153

(g) Seuils humains



(h) 64 000 échantillons

## Backup : Comparaisons GGN

### Comparaisons

- Modèle RNN avec entropie SVD
- Sur la même base d'apprentissage que le GGN
- Meilleure performance du modèle entropie SVD

# Backup : Comparaisons GGN

## Comparaisons

- Modèle RNN avec entropie SVD
- Sur la même base d'apprentissage que le GGN
- Meilleure performance du modèle entropie SVD

## Comparaison des seuils de classification :

t threshold		Accuracy Train	AUC ROC Train	Accuracy Test	AUC ROC Test	Accuracy Global	AUC ROC Global
RNN	0.3	0.7619	0.9115	0.8122	0.9297	0.7682	0.9137
	0.4	0.8085	0.9115	0.8456	0.9297	0.8131	0.9137
	0.5	0.8281	0.9115	<b>0.8519</b>	0.9297	0.8311	0.9137
	0.6	0.8329	0.9115	0.8385	0.9297	0.8336	0.9137
	...	...	...	...	...	...	...
GGN	...	...	...	...	...	...	...
	0.8	0.8809	0.9735	0.7709	0.8422	0.8672	0.9571
	0.9	0.9067	0.9735	0.7858	0.8422	0.8916	0.9571
	0.95	0.9204	0.9735	0.8013	0.8422	0.9055	0.9571
	0.98	0.9201	0.9735	<b>0.8216</b>	0.8422	0.9078	0.9571

## Modèle de perception : Population et résultats

### Population

- 17 participants
- 5 femmes et 12 hommes
- Âge moyen de 31 ans

## Modèle de perception : Population et résultats

### Population

- 17 participants
- 5 femmes et 12 hommes
- Âge moyen de 31 ans

Nombre d'images et pourcentage d'images considérées comme non bruitées :

	RNN - Entropie SVD	GGN
Images	403/476	419/476
Pourcentage	85%	88%

## Backup : Optimisation pseudo-booléenne

**Optimisation pseudo-booléenne :**

$$f : \{0, 1\}^n \rightarrow \mathbb{R}$$

- Taille de l'espace de recherche est  $2^n$

## Backup : Optimisation pseudo-booléenne

**Optimisation pseudo-booléenne :**

$$f : \{0, 1\}^n \rightarrow \mathbb{R}$$

- Taille de l'espace de recherche est  $2^n$

**Opérateurs de variation :**

- Mutation
- Permutation
- Croisement

## Backup : Optimisation pseudo-booleenne

**Optimisation pseudo-booleenne :**

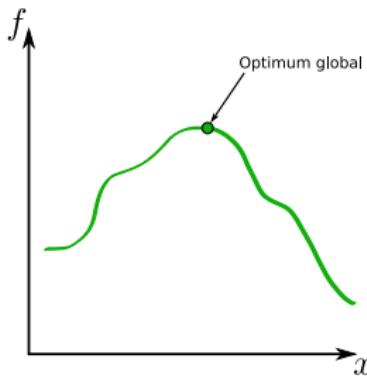
$$f : \{0, 1\}^n \rightarrow \mathbb{R}$$

- Taille de l'espace de recherche est  $2^n$

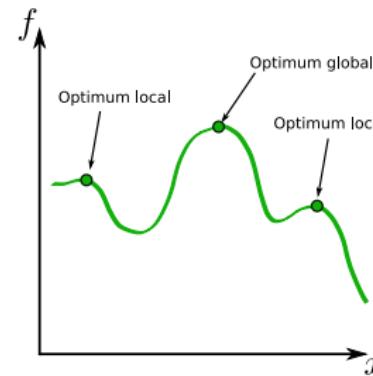
**Opérateurs de variation :**

- Mutation
- Permutation
- Croisement

**Compromis exploration et exploitation :**



(a) Espace de recherche simple



(b) Espace de recherche convexe

## Backup : Apprentissage du méta-modèle

Framework d'apprentissage pour l'utilisation de méta-modèle [Leprêtre et al., 2019]

---

**Result:** Meilleure solution trouvée  $x'$

```
1 initialiser:  $S \leftarrow$  solutions initiales  $\{(x, f(x)), \dots\}$ ;  
2 while critère d'arrêt do  
3   |    $M \leftarrow$  construire un méta-modèle  $M$  depuis l'ensemble  $S$  de solutions ;  
4   |    $x \leftarrow$  recherche locale en utilisant le méta-modèle  $M$  ;  
5   |   Évaluer  $x$  en utilisant  $f$  ;  
6   |   Mettre à jour  $x'$  en fonction de  $x$  ;  
7   |    $S \leftarrow S \cup \{x, f(x)\}$  ;  
8 end  
9 return  $x'$ ;
```

---

## Backup : Paramètres étudiés

### Jeux de paramètres

- Le nombre d'évaluations réel maximal à 1 000
- Le nombre d'évaluation par recherche locale avec le méta-modèle,  $LS \in [100, 500, 1000]$
- L'ordre de Walsh  $L \in [1, 2]$

### Gestion des opérateurs de variation

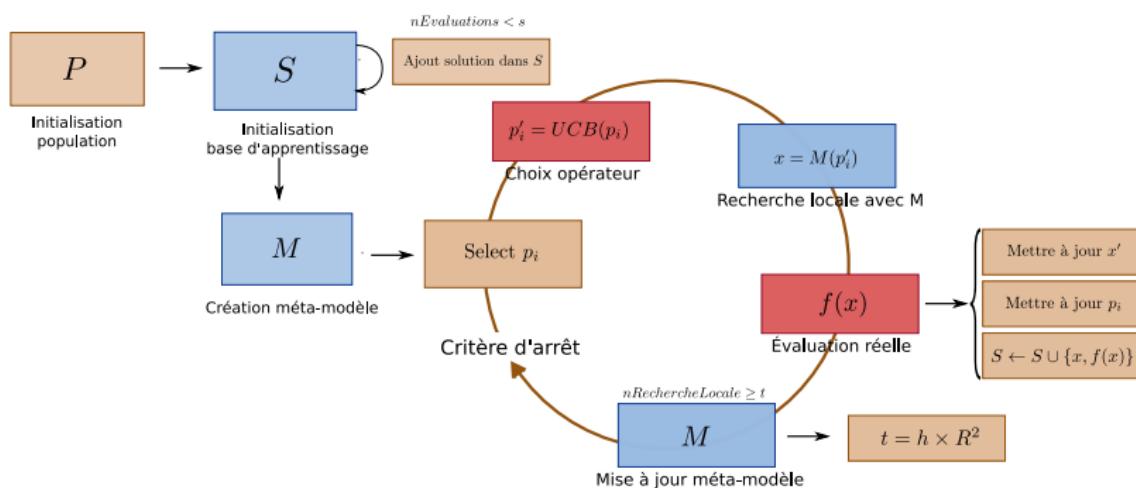
#### Liste des opérateurs :

- Un opérateur de mutation
- Un opérateur de permutation
- Un opérateur de croisement centré
- Un opérateur de croisement aléatoire

#### Choix des opérateurs :

- Utilisation de l'algorithme *Upper Confidence Bound* [Liu et al., 2017]

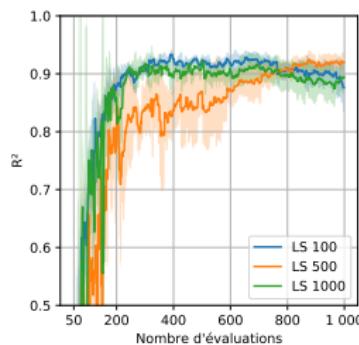
## Backup : Méta-heuristique utilisée



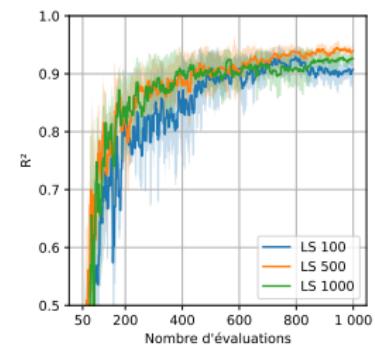
## Backup : Apprentissage modèle

### Performance d'apprentissage du méta-modèle :

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - y'_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$



(a)  $L = 1$

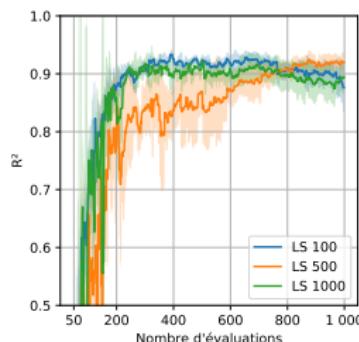


(b)  $L = 2$

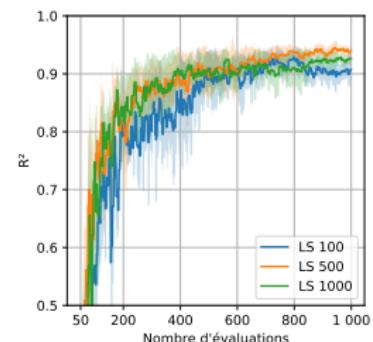
## Backup : Apprentissage modèle

### Performance d'apprentissage du méta-modèle :

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - y'_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$



(a)  $L = 1$

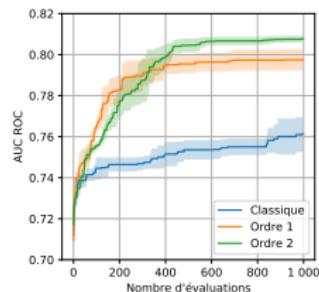
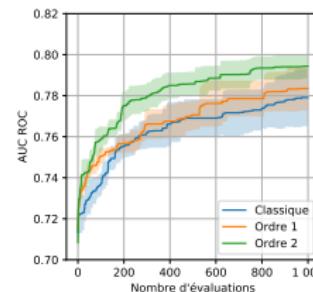
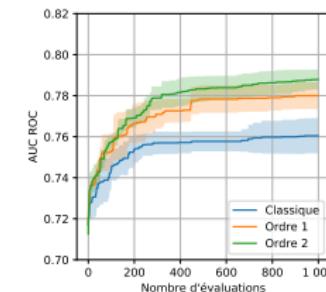


(b)  $L = 2$

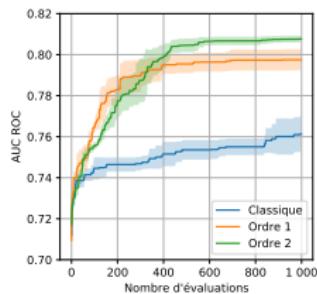
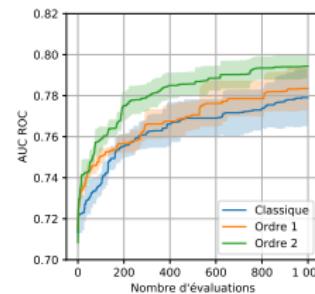
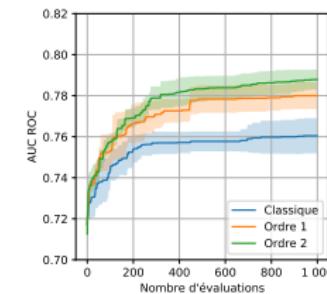
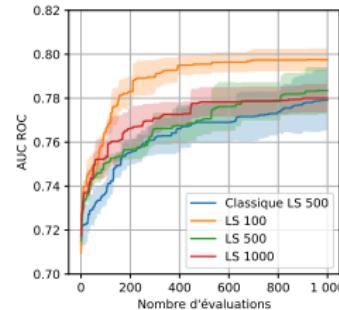
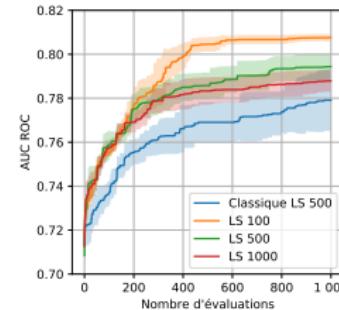
### Temps moyens :

	Classique	$L = 1$	$L = 2$
Heures	6,05	6,228	13,667

## Backup : Résultats obtenus

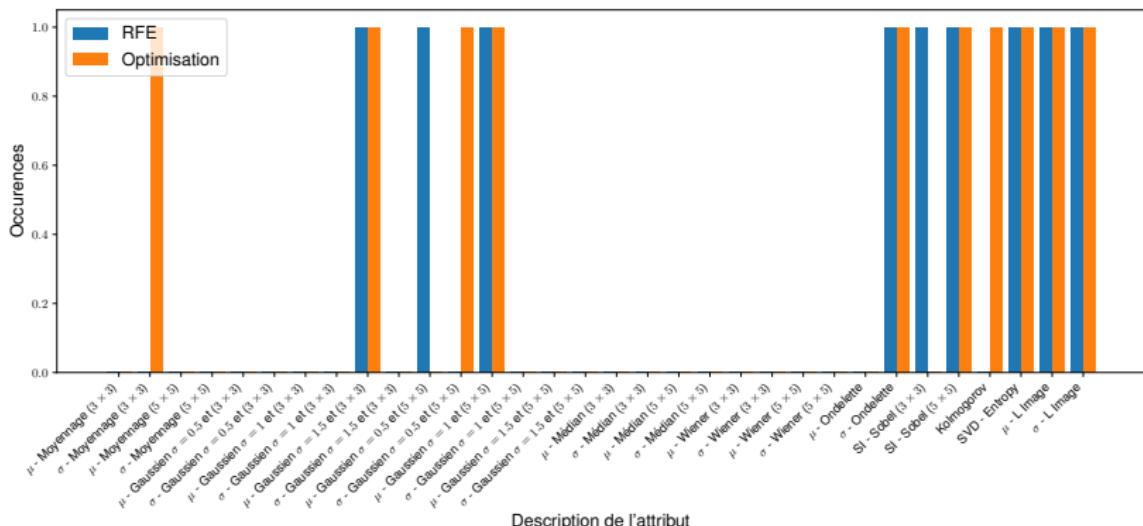
Performance de l'ordre de Walsh  $L$  :(a)  $LS = 100$ (b)  $LS = 500$ (c)  $LS = 1\,000$

## Backup : Résultats obtenus

Performance de l'ordre de Walsh  $L$  :(a)  $LS = 100$ (b)  $LS = 500$ (c)  $LS = 1\,000$ Impact du paramètre  $LS$  :(a)  $L = 1$ (b)  $L = 2$

# Backup : Sélection d'attributs comparaisons

## Comparaison des attributs sélectionnés avec la Méthode RFE :



## Attributs sélectionnés pour l'optimisation

Modèle avec le meilleur score AUC ROC : 10 attributs sélectionnés