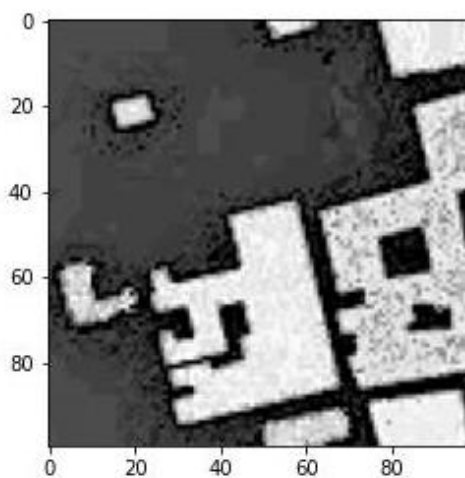


Projet de Fin d'Études (PFE) 2022-2023

Intelligence Artificielle et Potentiel Solaire



L'intelligence artificielle et le potentiel solaire en milieu urbain

*Comparaison d'un algorithme d'apprentissage
automatique avec un algorithme de simulation pour le
calcul du potentiel solaire en milieu urbain*

**Directeur de recherche
Mindjid Maïzia**

**Auteur
Paul Vola**

2022/2023

AVERTISSEMENT

Cette recherche a fait appel à des lectures, enquêtes et interviews. Tout emprunt à des contenus d'interviews, des écrits autres que strictement personnel, toute reproduction et citation, font systématiquement l'objet d'un référencement.

L'auteur (les auteurs) de cette recherche a (ont) signé une attestation sur l'honneur de non plagiat.

Formation par la recherche, Projet de Fin d'Etudes en génie de l'Aménagement et de l'Environnement

La formation au génie de l'aménagement et de l'environnement, assurée par le département aménagement et environnement de l'Ecole Polytechnique de l'Université de Tours, associe dans le champ de l'urbanisme, de l'aménagement des espaces fortement à faiblement anthropisés, l'acquisition de connaissances fondamentales, l'acquisition de techniques et de savoir faire, la formation à la pratique professionnelle et la formation par la recherche. Cette dernière ne vise pas à former les seuls futurs élèves désireux de prolonger leur formation par les études doctorales, mais tout en ouvrant à cette voie, elle vise tout d'abord à favoriser la capacité des futurs ingénieurs à :

- Accroître leurs compétences en matière de pratique professionnelle par la mobilisation de connaissances et de techniques, dont les fondements et contenus ont été explorés le plus finement possible afin d'en assurer une bonne maîtrise intellectuelle et pratique,
- Accroître la capacité des ingénieurs en génie de l'aménagement et de l'environnement à innover tant en matière de méthodes que d'outils, mobilisables pour affronter et résoudre les problèmes complexes posés par l'organisation et la gestion des espaces.

La formation par la recherche inclut un exercice individuel de recherche, le projet de fin d'études (P.F.E.), situé en dernière année de formation des élèves ingénieurs. Cet exercice correspond à un stage d'une durée minimum de trois mois, en laboratoire de recherche, principalement au sein de l'équipe Dynamiques et Actions Territoriales et Environnementales de l'UMR 7324 CITERES à laquelle appartiennent les enseignants-chercheurs du département aménagement.

Le travail de recherche, dont l'objectif de base est d'acquérir une compétence méthodologique en matière de recherche, doit répondre à l'un des deux grands objectifs :

- Développer toute ou partie d'une méthode ou d'un outil nouveau permettant le traitement innovant d'un problème d'aménagement
- Approfondir les connaissances de base pour mieux affronter une question complexe en matière d'aménagement.

Afin de valoriser ce travail de recherche nous avons décidé de mettre en ligne sur la base du Système Universitaire de Documentation (SUDOC), les mémoires à partir de la mention bien.

ABSTRACT

Les Méthodes de simulation, comme le Ray Tracing, permet un calcul précis du potentiel solaire dans un milieu urbain. Mais cette technique demande un temps de calcul important de l'ordre de plusieurs heures. Afin de déterminer les zones d'installation de panneaux photovoltaïques à fort potentiel il est important de disposer d'une méthode de calcul précise et rapide. Depuis plusieurs années les algorithmes d'intelligence artificielle sont utilisés dans de nombreux domaines dont celui de l'analyse d'images. Lors de ce projet de fin d'étude, j'ai développé un algorithme basé sur le Machine Learning permettant de déterminer le potentiel solaire en zone urbaine, puis je l'ai comparé avec la méthode de Ray Tracing sur les critères de la précision des résultats et celui du temps de calcul. Les résultats de cette comparaison montrent que la précision est comparable mais que les temps de calcul sont environ 100 fois plus rapide en faveur de mon algorithme (à puissance égale).

REMERCIEMENTS

Je tiens tout d'abord à remercier Monsieur Mindjid Maïzia, mon tuteur pédagogique pour ce projet de fin d'études (PFE) et directeur de l'option RESEAU (Réseaux, Énergies et Systèmes de l'Aménagement Urbain) au sein du département Environnement et Aménagement de Polytech Tours. M Maïzia m'a accompagné tout au long de ce projet en étant toujours disponible. Je le remercie de m'avoir aidé à comprendre le projet, pour ses remarques et son aide dans la construction de l'algorithme.

Je remercie aussi mon grand-père, ma maman et mon papa qui m'ont apporté leur aide dans la rédaction de ce projet par leurs remarques pertinentes, et en me relisant.

J'ai aussi un petit mot pour mon camarade Alexandre Bouzaires qui a travaillé sur le même sujet, avec qui j'ai échangé ce qui nous a permis de confronter nos travaux et de réaliser un travail plus complet.

Sommaire :

Abstract :	6
Remerciements :	7
Sommaire :	8
Introduction :	9
Enjeux : Déterminer le potentiel solaire en milieu urbain :	10
Hypothèse :	11
1. Le Ray Tracing :	12
1.1. Le Principe du Ray Tracing :	12
1.2. Le Ray tracing et le potentiel solaire :	13
2. Intelligence Artificielle et algorithmique :	15
2.1. Définition de l'Intelligence Artificielle (IA) :	15
2.2. Le Machine Learning :	16
2.3. Mon algorithme d'Intelligence Artificielle pour calculer le potentiel solaire :	16
3. Résultat, Comparaison des deux méthodes :	20
3.1. Les paramètres étudiés :	20
3.2. Conditions expérimentales :	20
3.3. Temps de calcul :	20
3.4. Précision :	21
Conclusion :	24
Bibliographie :	25
Annexe :	26
Résultat :	26
Construction de l'algorithme	42
Code de l'algorithme :	44
Méthode du Ray Tracing :	52
Tables de figure :	57

Introduction :

Le soleil représente une source d'énergie propre et abondante. Pour avoir un mix énergétique bas carbone, il est nécessaire d'installer des panneaux photovoltaïques (PV) en milieu urbain. Mais l'installation de ces panneaux photovoltaïques demande en amont de déterminer le potentiel solaire des surfaces capables d'accueillir ces panneaux. Cela permet ainsi d'identifier les zones les plus favorables c'est-à-dire celles qui produiront le plus d'énergie. Pour réaliser ces études il existe une multitude de méthodes basées sur des techniques plus ou moins précises. Nous pouvons en citer deux, la simulation de l'exposition au soleil et des méthodes statistiques. Depuis les années 2000, l'apprentissage automatique « Machine Learning » est utilisé dans de nombreux domaines (économie, biologie, surveillance, art...), il a aussi été mis en avant dans le domaine de l'analyse d'images. C'est une méthode nouvelle que l'on peut utiliser dans l'étude du potentiel solaire. Ce projet de fin d'étude (PFE) a pour objectif de développer un algorithme permettant de déterminer le potentiel solaire en utilisant le Machine Learning et de comparer cet algorithme aux autres méthodes plus classique comme celle du Ray Tracing. Dans ce rapport vous trouverez, la description de la méthode de Ray Tracing et celle de notre méthode basée sur l'intelligence artificielle ainsi qu'une comparaison de ces deux méthodes.

Enjeux : Déterminer le potentiel solaire en milieu urbain :

L'énergie est la base des civilisations industrielles et conditionne notre mode de vie. Pour produire de l'Énergie, nous utilisons majoritairement des ressources fossiles (charbon, produit pétrolier, gaz naturel) 81 % du mix énergétique mondial en 2018. En France, le mix énergétique est composé de 46 % de pétrole, gaz et charbon, de 40 % de nucléaire, et 14 % d'énergie renouvelable¹. Cette production d'énergie est polluante, il est indispensable de développer les énergies renouvelables pour lutter contre le dérèglement climatique.

Le soleil représente une importante source d'énergie propre que nous pouvons d'exploiter par l'installation de panneaux photovoltaïques. En France, l'ADEME estime qu'il y a 2276 kilomètres carrés de toiture. Une partie de cette surface peut accueillir des panneaux photovoltaïques sans faire diminuer d'autre surfaces importantes comme par exemple les fermes solaires qui réduisent la surface agricole. Pour cela, on doit déterminer la quantité d'énergie reçue par la surface de toit. Si celle-ci est supérieure à 1 000 kWh/m²/an alors il est judicieux d'y installer des panneaux photovoltaïques.

Le calcul du potentiel solaire en milieu urbain est complexe, car il y a des zones d'ombre, de la réflexion... Il existe plusieurs techniques pour calculer le potentiel solaire dans un milieu urbain. Les plus utilisées sont les techniques de simulation, comme le Ray Tracing. Le Ray Tracing est une méthode précise mais qui a un temps de calcul important.

¹ *Bilan énergétique de la France*. (s. d.). Chiffres clés de l'énergie - Édition 2021.
<https://www.statistiques.developpement-durable.gouv.fr>

Hypothèse :

Depuis les années 2000, les intelligences artificielles sont de plus en plus utilisées pour leurs performances notamment dans la reconnaissance d'image et dans la réduction du temps de calcul des simulations. Ces nouvelles techniques montrent de très bonnes performances et une rapidité de calcul supérieure aux algorithmes plus classiques.

On fait l'hypothèse que l'utilisation d'un algorithme de Machine Learning pour déterminer le potentiel solaire est plus rapide et produit des résultats comparables à une méthode de Ray Tracing.

1. Le Ray Tracing :

Le Ray Tracing est un procédé informatique qui consiste à créer une image (en deux dimensions) selon un point de vue en simulant les effets de la lumière sur des objets en trois dimensions.

Le premier code de Ray Tracing a été inventé par Arthur Appel en 1968. Mais les premières utilisations furent au début des années 2000 dans les films d'animation. Puis le procédé a été utilisé dans les jeux vidéo, l'animation 3D...

1.1. Le Principe du Ray Tracing :

Le Ray Tracing, utilise les lois de l'optique géométrique de la lumière. Notamment le principe de Fermat et les lois optiques de Snell-Descartes sur le comportement des rayons lumineux pour simuler la géométrie des rayons lumineux.

- Le principe de Fermat ou Principe de retour vers la lumière dit « Le trajet suivi par la lumière pour aller d'un point à un autre ne dépend pas du sens de propagation de la lumière. ». En s'appuyant sur ce principe, on simule les rayons lumineux depuis le point d'observation et non depuis les sources lumineuses.
- Les lois de Snell-Descartes définissent le comportement d'un rayon lumineux à l'interface de deux milieux.

Réflexion :

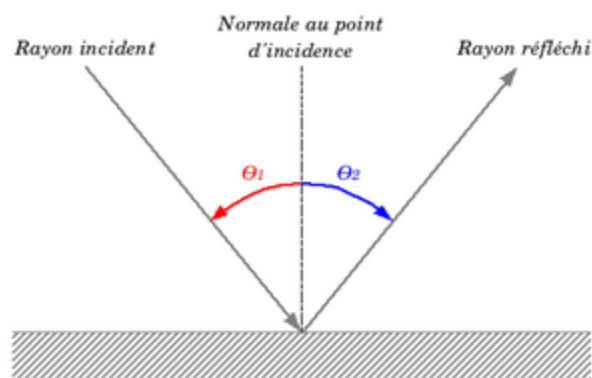


Figure 1 : Schéma de la réflexion, le faisceau incident va être dévié selon la loi dite de Snell-Descartes

Source : Wikipédia

Lorsqu'un rayon rencontre un objet, l'angle entre le rayon incident et la normale du point de rencontre de l'objet est le même qu'entre la normale et le rayon réfléchi.

Réfraction :

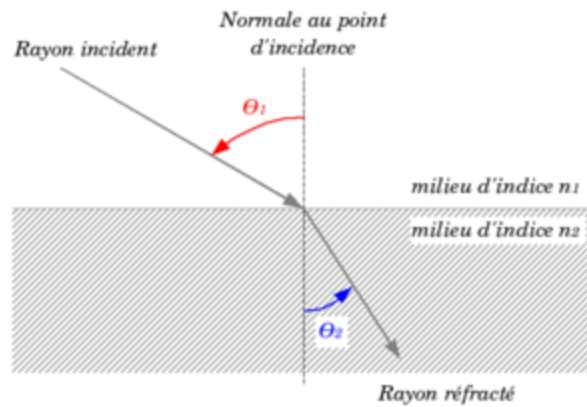


Figure 2 : Schéma de la réfraction, le faisceau incident va être dévié selon la loi dite de Snell-Descartes

Source : Wikipédia

Dans le cas d'une réfraction, la géométrie des rayons incident et réfracté suit la loi suivante, $n_1 \sin \theta_1 = n_2 \sin \theta_2$

Le Ray Tracing consiste à générer un rayon lumineux depuis le point de vue (la caméra) et passant par un pixel jusqu'à un point de rencontre, puis à projeter des rayons secondaires vers les sources lumineuses et les autres surfaces pour déterminer la luminosité.

Ce procédé est réalisé pour chaque pixel afin d'obtenir une image avec les éclairages.

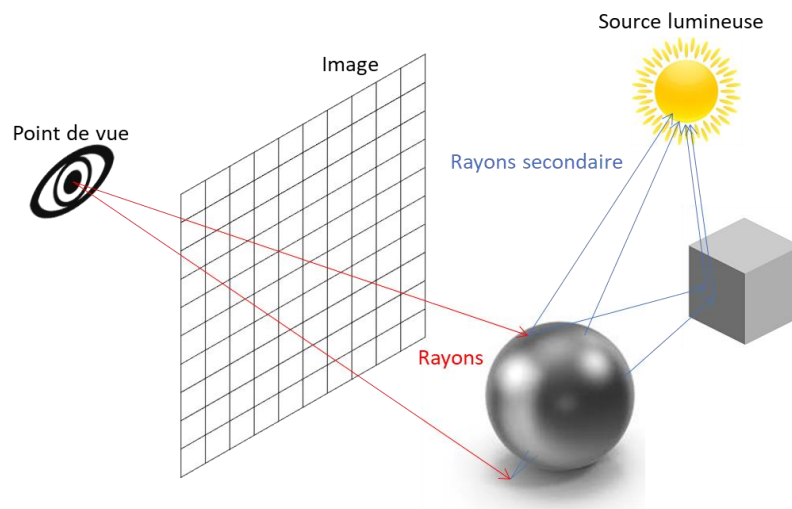


Figure 3 : Schéma présentant le Ray Tracing et le chemin des rayons simulés

Source : réalisation personnelle

1.2. Le Ray tracing et le potentiel solaire :

Il est possible d'utiliser le Ray tracing pour déterminer les zones d'ensoleillement et d'ombrage dans un milieu urbain.

On commence par modéliser un milieu urbain, puis on génère une image aérienne avec le procédé de Ray Tracing et une source lumineuse similaire au soleil durant une année. Et on identifie les zones

avec un fort ensoleillement. Il est possible de réaliser cette technique grâce à SketchUp et son extension Twilight.

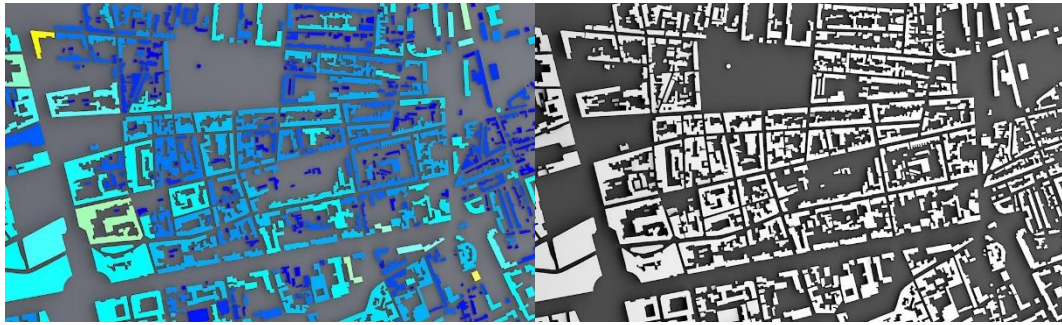


Figure 4 : Exemple de vue aérienne des bâtiments modélisés (figure de gauche) et de rendu du potentiel solaire (figure de droite) obtenus par Ray Tracing à partir du logiciel SketchUp et de l'extension Twilight.

Source : réalisation personnelle

Comme nous l'avons vu, pour réaliser une image des milliers voire des millions de rayons doivent être simulés. Ainsi le Ray Tracing est un procédé long qui demande une grande puissance de calcul. Les industries du cinéma et du jeu vidéo n'ont commencé à l'utiliser que récemment car beaucoup trop onéreux jusqu'il y a peu.

2. Intelligence Artificielle et algorithmique :

2.1.Définition de l'Intelligence Artificielle (IA) :

« Ensemble de théories et de techniques mises en œuvre en vue de réaliser des machines capables de simuler l'intelligence humaine. » Larousse

2.1.1. Les premières idées d'intelligence artificielle :

On trouve l'idée de machines capables de penser depuis très longtemps. Les premières traces remontent à l'antiquité dans la mythologie grecque. Les exemples les plus connus sont : Pandore (une créature sculptée par les dieux à laquelle ils ont donné la capacité de penser) et les servantes en or Héphaïstos.

L'idée d'une machine capable de penser a toujours fait fantasmer, puisqu'on la retrouve jusqu'à aujourd'hui dans les romans (Frankenstein ou le Prométhée moderne – 1818) et dans les films. (Blade Runner – 1989)

À partir de ces deux définitions, on peut caractériser l'intelligence artificielle par une machine ou un programme informatique qui visent à imiter la réflexion humaine. L'intelligence artificielle se fonde sur l'hypothèse que le processus de pensée humaine peut être mécanisé.

2.1.2. 1950, naissance officielle de l'intelligence artificielle :

En 1950, Alan Turing et John Von Neumann vont poser les prémisses de l'intelligence artificielle sous forme technologique. Ces derniers ont créé les premières architectures d'ordinateurs avec de la programmation et les utiliser pour résoudre des problèmes. Dans un article « Computing Machinery and Intelligence », Alan Turing va parler d'une machine capable d'imiter l'humain et proposer un test pour déterminer quand une machine devient consciente, le Test de Turing.

La conférence au Dartmouth Collège en 1956 est considérée comme la naissance officielle de l'intelligence artificielle. Durant cette conférence, Marvin Lee Minsky définira l'IA comme : « La construction de programmes informatiques qui s'adonnent à des tâches qui sont, pour l'instant, accomplies de façon plus satisfaisante par des êtres humains car elles demandent des processus mentaux de haut niveau tels que : l'apprentissage perceptuelle, l'organisation de la mémoire et le raisonnement critique »².

À la suite de cette conférence, les premiers arbres de recherche de solution vont être proposés. Mais les IA vont avoir du mal à se développer du fait du manque de mémoire des ordinateurs.

2.1.3. Le développement des IA :

En 1971, les premiers microprocesseurs permettent d'élaborer des algorithmes plus complexes et plus puissants, une vraie révolution. Ce qui relance les recherches en Intelligence Artificielle. Mais la concurrence avec des méthodes moins onéreuses va ralentir le développement des IA.

À partir des années 2000, on va connaître une nouvelle évolution grâce aux Big Data et des processeurs plus puissants. On ne cherche plus à « apprendre » aux algorithmes, mais on les laisse apprendre à partir de jeux de données, c'est le Machine Learning.

² , Marvin Lee Minsky, Conférence au Dartmouth Collège (1956)

2.1.4. Les applications aujourd'hui :

Aujourd'hui, l'intelligence artificielle est utilisée dans nombreux domaines comme les échecs, l'économie, la biologie, la robotique, le marketing, ...

2.2. Le Machine Learning :

2.2.1. Définition :

Yann LeCun, chercheur en intelligence artificielle et un des inventeurs de l'apprentissage profond définit le Machine Learning « apprentissage automatique » comme : « une catégorie de programme informatique simple, mais avec de nombreux paramètres qu'on ne peut définir. Ces paramètres sont déterminés par le programme en faisant multiples de calculs »³.

Le Machine Learning est donc une sous-catégorie d'intelligence artificielle. C'est un algorithme à qui l'on va faire étudier une grande quantité de données pour qu'il trouve des liens et des corrélations (des patterns) pour ensuite faire des prédictions.

2.2.2. Fonctionnement :

Le fonctionnement d'un algorithme de Machine Learning se fait en deux temps.

- Le premier est la phase dite « d'entraînement ». On donne à notre algorithme un jeu de données dont on connaît le résultat. Et il va devoir déterminer les corrélations entre les données pour atteindre le bon résultat.
- La deuxième phase est la prédiction. On donne à notre algorithme des données similaires aux données d'entraînements dont on ne connaît pas le ou les résultats. Et l'algorithme fait une prédiction du résultat.

2.2.3. Les différentes catégories de Machine Learning :

Il existe une multitude d'algorithmes de Machine Learning. On peut les catégoriser selon leur mode d'apprentissage (supervisé, non-supervisé, semi-supervisé, partiellement supervisé...), soit en fonction de leurs architectures (régression, réseau de neurones, arbre de décisions, méthode des plus proches voisins...). Il est important de bien choisir quel algorithme de Machine Learning utiliser en fonction de son projet.

2.2.4. Les Applications

Les algorithmes de Machine Learning sont utilisés pour traiter de grandes quantités de données. Les applications principales sont : la reconnaissance, la classification, la détection, la prédiction, ou la diminution de temps de calcul de simulation.

2.3. Mon algorithme d'Intelligence Artificielle pour calculer le potentiel solaire :

Dans le cadre de ce projet de fin d'études à Polytech Tours, j'ai développé un algorithme qui a pour but de générer des images montrant le potentiel solaire. Cet algorithme utilise le module « KNeighborsClassifier ».

³ Yann LeCun, « The Conversation », *Autour de l'informatique. Yann LeCun : l'apprentissage profond avant tout.* (10 novembre 2017).

2.3.1. Méthode des K plus proches, classification :

La méthode des k plus proches est une méthode d'apprentissage supervisé. C'est-à-dire que les données doivent être étiquetées. Si c'est un algorithme de classification l'étiquetage est discret (des nombres entiers, une liste, des noms d'animaux, des formes...)

L'algorithme stock une base de référence qui correspond à la base d'apprentissage. Cette base est constituée de données avec des valeurs en entrées et une classe de sortie.

Pour prédire une classe de sortie, on entre une nouvelle donnée dans le même format que les données de paramétrage. L'algorithme cherche les k donnés avec les entrées les plus proches dans la base d'apprentissage. Et le résultat est la classe de sortie majoritaire dans les k les plus proches.

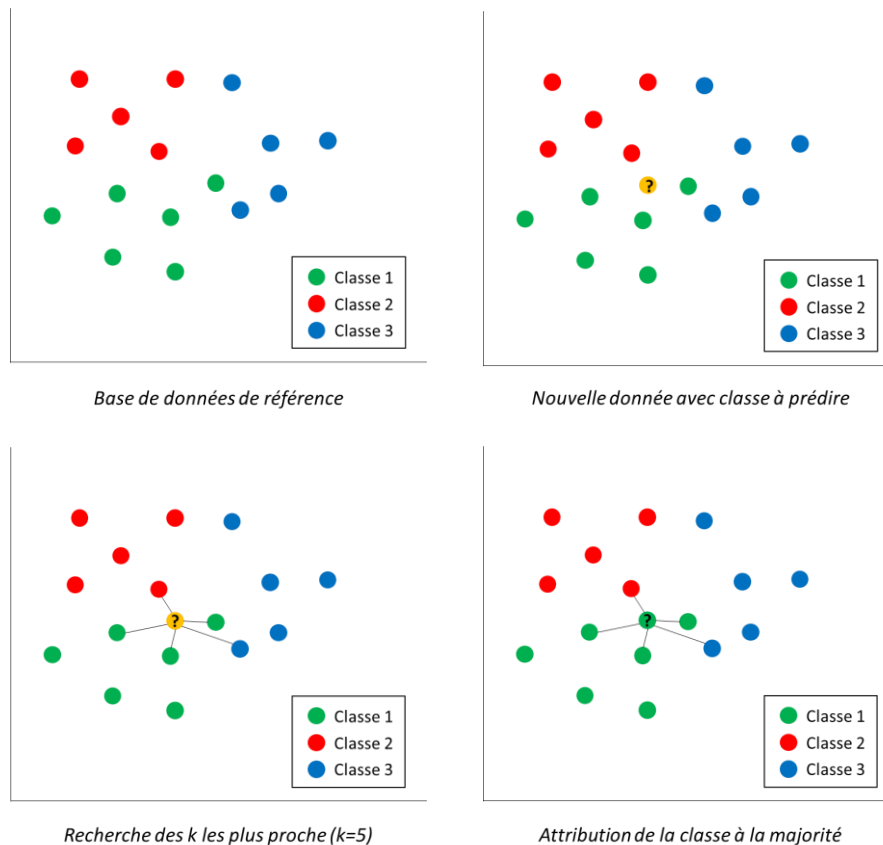


Figure 5 : Les étapes de la méthode d'apprentissage automatique des k plus proches

Source : Réalisation personnelle

Avec la méthode des k plus proches, il y a peu de paramètres à définir. Le premier à définir est k, il est recommandé que k soit un nombre premier afin de limiter les égalités. k dépend aussi de la base de données d'apprentissage. Si celle-ci comprend des valeurs aberrantes, il faudra un k élevé pour diminuer l'influence de ces valeurs. Mais un k élevé demande un temps et une puissance de calcul plus importante. Pour mon algorithme, j'ai choisi un k égal à 5.

Le second paramètre important est la méthode de calcul des distances. Il existe plusieurs façons de mesurer les distances, les principales sont :

Distance Euclidienne : $d(x, y, \dots, z) = \sqrt{\sum (x_0 - x_i)^2 + (y_0 - y_i)^2 + \dots + (z_0 - z_i)^2}$

Distance Manhattan : $d(x, y, \dots, z) = \sum |x_0 - x_i| + |y_0 - y_i| + \dots + |z_0 - z_i|$

Distance Minkowski : $d(x, y, \dots, z) = (\sum |x_0 - x_i| + |y_0 - y_i| + \dots + |z_0 - z_i|)^{1/p}$

Pour mon algorithme j'ai choisi la distance Euclidienne.

L'algorithme d'apprentissage automatique que j'ai utilisé pour mon algorithme est le module « KNeighborsClassifier » de la bibliothèque Scikit-Learn.

Scikit-Learn est une des bibliothèques Python d'apprentissage automatique libre de droit parmi les plus connues et utilisées. Elle propose plusieurs types d'algorithmes d'apprentissage automatique (classification, régression, regroupement, réduction de dimensions, prétraitement et sélection de modèles).

2.3.2. Fonctionnement et utilisation de l'algorithme :

L'algorithme prend en entrée une image aérienne des bâtiments avec pour chaque bâtiment une couleur correspondant à sa hauteur. Puis il génère une image en vue aérienne du potentiel solaire associé aux bâtiments.

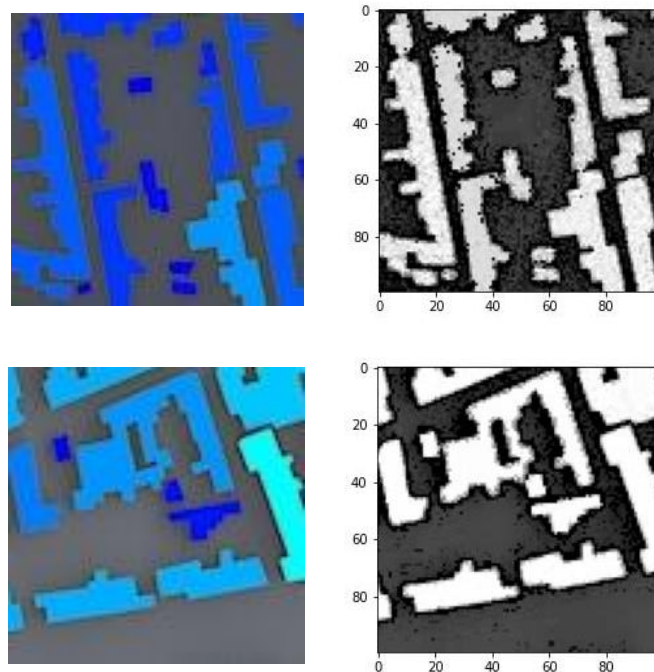


Figure 6 : Exemples d'images du potentiel solaire réalisées grâce à l'algorithme développé dans le cadre du PFE

Source : Réalisation personnelle

Pour l'apprentissage de l'algorithme, j'ai utilisé des images en vue aérienne du potentiel solaire obtenues grâce à la méthode de Ray Tracing.

2.3.3. Données d'entrées :

Notre algorithme prend en donnée d'entrée, une image aérienne des bâtiments avec pour chaque bâtiment une couleur correspondant à sa hauteur.

Cette image aérienne est obtenue à partir de la cartographie des bâtiments que l'on peut retrouver dans les bases de données IGP (format Shapefile). Puis grâce à l'application Toasterintegral Plus, on

modélise les bâtiments en associant leurs couleurs à leurs hauteurs, que l'on peut importer dans SketchUp. Enfin, une photographie aérienne du milieu urbain est réalisée avec l'extension Twilight.

3. Résultat, Comparaison des deux méthodes :

Il convient de comparer l'algorithme développé lors de ce PFE avec le procédé de Ray Tracing dans le cadre de détermination du potentiel solaire en milieu urbain.

3.1. Les paramètres étudiés :

Le but est de déterminer si utiliser un algorithme d'apprentissage automatique est plus efficient qu'une simulation par Ray Tracing.

Nous comparons les temps de calcul des deux méthodes, et les résultats de l'apprentissage automatique par rapport au Ray Tracing.

3.2. Conditions expérimentales :

Il est nécessaire de générer des images de potentiel solaire générées grâce à notre algorithme et à la technique de Ray Tracing issues du même environnement urbain (la ville de Tours). Ainsi, pour toute image obtenue par notre algorithme on a la même image obtenue par le procédé du Ray Tracing.

Pour le Ray Tracing, nous allons utiliser SketchUp et son extension Twilight suivant la méthode décrite en annexe (annexe n°8).

Les images ont été générées avec le même ordinateur et sous le même format afin d'être dans les mêmes conditions expérimentales.

3.3. Temps de calcul :

3.3.1. Comparaison des deux méthodes :

J'ai généré 307 images de 100 pixels par 100 pixels en utilisant l'algorithme développé lors de ce PFE. Pour ces 307 images, le temps pour générer une image est compris entre 0,817 et 3,029 secondes avec une moyenne de 1,226 et une médiane de 1 seconde (ensemble des données en annexes).

	Moyenne	Minimal	Maximale	Médiane
Temps (en sec)	1,222	0,817	3,029	1,000

Figure 7 : Tableau récapitulatif des 307 images de potentiel solaire réalisé avec l'algorithme du PFE

Source : Réalisation personnelle

Avec la Méthode de Ray Tracing, j'ai généré 15 images de 100 pixels par 100 pixels. En moyenne, l'image est générée en 1 minute 58.

Donc, en moyenne, l'algorithme de Machine Learning est 100 fois plus rapide que l'algorithme de Ray Tracing.

On observe aussi que le temps de production d'une image avec du Ray Tracing dépend à la fois de la taille de l'image et de la complexité de l'environnement 3D. Des différences de temps de l'ordre d'un facteur 4 entre des images simples avec un seul bâtiment (55 secondes) et des images avec 100 bâtiments (2 minutes 58).

Notre algorithme a un temps de calcul plutôt constant pour des images de mêmes tailles. Les différences de temps de calcul sont dues à la machine.

Utiliser un algorithme d'apprentissage automatique est donc 100 fois plus rapide qu'une méthode de Ray Tracing. On note aussi que le temps de calcul du Ray Tracing augmente avec la complexité de l'environnement ce qui n'est pas le cas de notre algorithme.

3.3.2. Durée en fonction de la taille :

On a fait fonctionner notre algorithme avec des images de tailles différentes. On obtient les résultats du graphe ci-dessous.

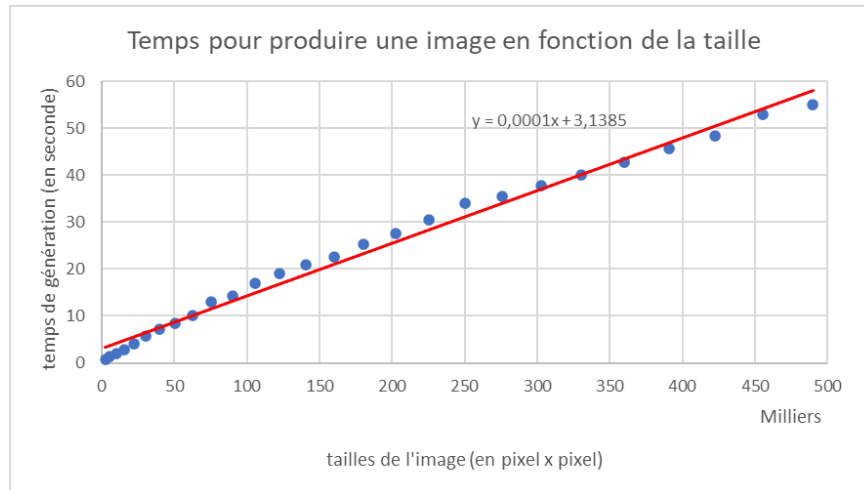


Figure 8 : Graphique représentant le temps pour produire une image en fonction de sa taille

Source : Réalisation personnelle

On peut voir que la durée pour générer une image de potentiel est proportionnelle à la taille de l'image. Ce qui confirme que la taille de l'image est le seul facteur dans la durée de calcul. D'après les résultats obtenus, il faut une seconde tous les 10 000 pixels.

$$\text{Durée}(\text{sec}) = \frac{\text{nb pixels}}{10\,000}$$

3.4. Précision :

Pour évaluer la précision de l'algorithme, on compare l'image produite avec l'image de référence obtenue par le Ray Tracing. La comparaison se fait selon le coefficient de corrélation de Pearson.

3.4.1. Le Coefficient de corrélation de Pearson (PCC) :

C'est un indice qui donne l'intensité d'une relation entre deux jeux de valeurs. On réalise un nuage de points avec en abscisse le premier jeu de données et en ordonnées le deuxième jeu de données en faisant correspondre les valeurs. Dans notre cas, on a en abscisse les valeurs RVB de l'image obtenue par Ray Tracing et en ordonnée les valeurs RVB de l'image obtenue grâce à notre algorithme, avec les valeurs en abscisse et en ordonnée correspondant au même pixel.

A partir de ce nuage de points, on obtient une droite linéaire. Puis on calcul la dispersion des points par rapport à la droite ce qui nous donne le coefficient de corrélation, avec la formule suivante :

$$r = \frac{\sum (x - m_x)(y - m_y)}{\sqrt{\sum (x - m_x)^2 \sum (y - m_y)^2}}$$

Avec m_x et m_y les valeurs de x et de y de la droite linéaire.

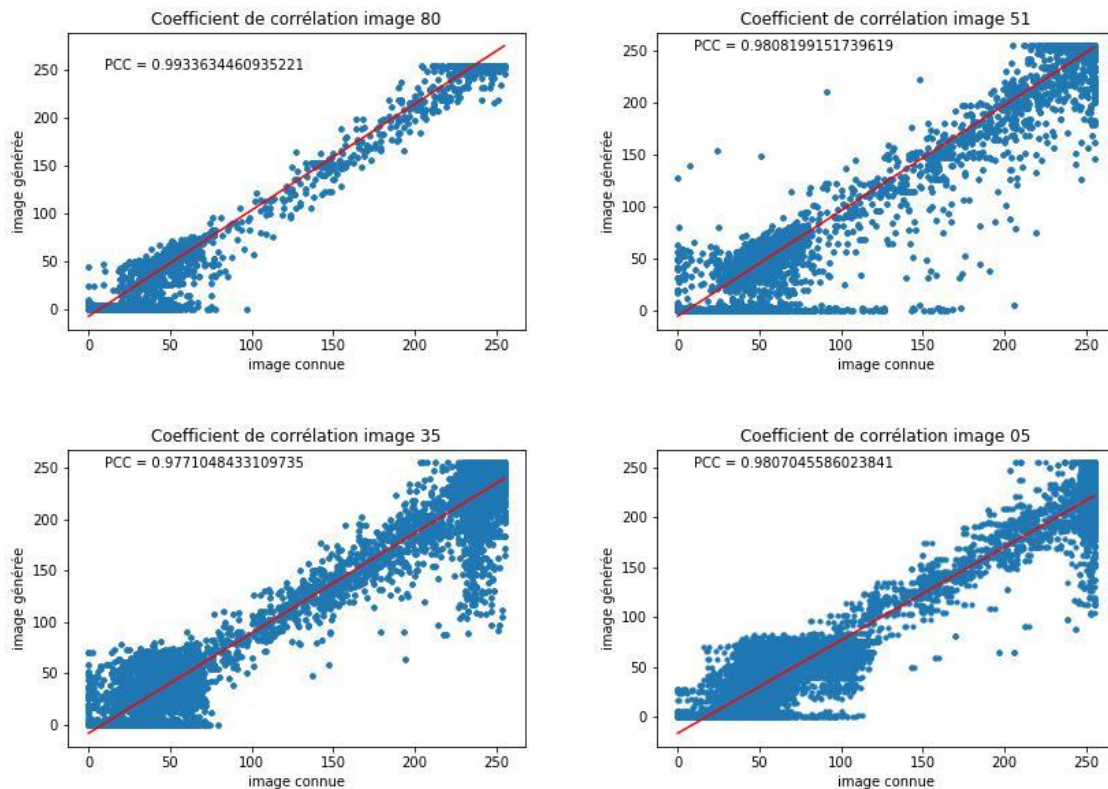


Figure 9 : Exemple de nuages de points représentant la corrélation entre les images obtenues par Ray Tracing et par apprentissage automatique

Source : Réalisation personnelle

Pour les 307 images générées, on obtient un coefficient de corrélation de Pearson moyen de 0,949.

Ce résultat est très bon puisque l'on est proche de 1. Il montre que les images obtenues avec notre algorithme sont quasiment similaires aux images obtenues avec le Ray Tracing.

3.4.2. Droite de régression :

La droite de régression est tracée à partir du nuage de points. Elle représente le lien entre les variables x et y.

Dans notre cas, on voudrait que les valeurs en abscisse soient identiques aux valeurs en ordonnée. Ce qui signifie que l'image générée avec notre algorithme est identique à l'image obtenue avec le Ray Tracing (avec un coefficient de Pearson de 1). Donc la droite idéale est $y = 1x + 0$ soit $y = x$.

En étudiant les 307 images générées, on obtient une droite de régression moyenne d'équation :

$$y = 0,971x - 10,965$$

Cette équation de droite est un très bon résultat. En effet 0,971 est très proche de 1, et -10,965 est aussi un bon résultat car on est sur une échelle comprise entre 0 et 255. Il n'est donc pas nécessaire d'appliquer une correction à notre algorithme.

3.4.3. Répartition des points :

Si l'on observe les nuages de points, on remarque que les points sont principalement concentrés aux extrémités autour des valeurs 0 et 255.

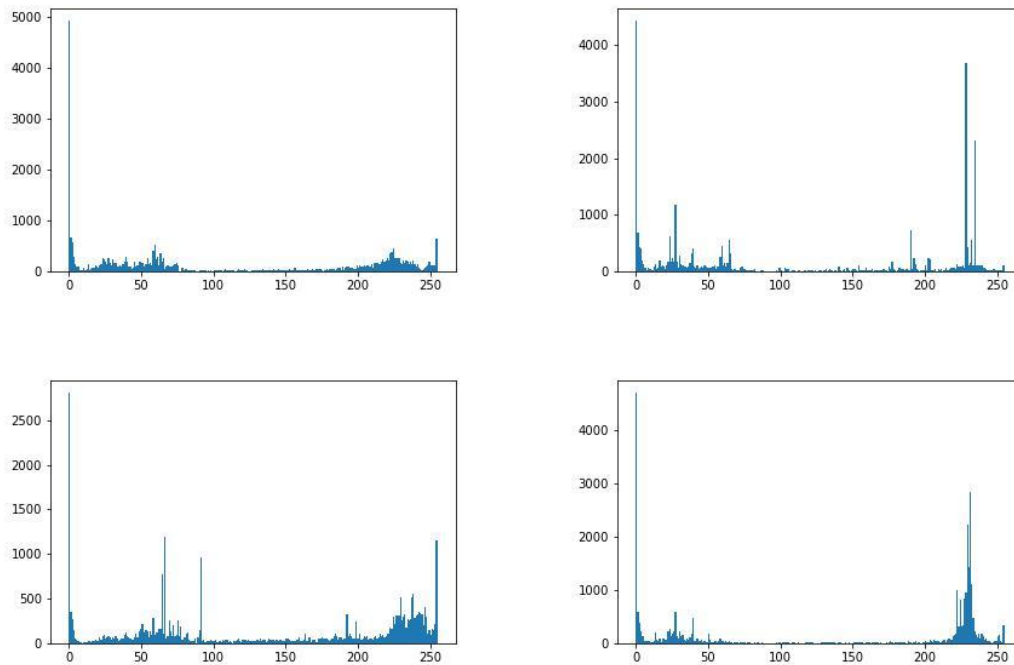


Figure 10 : Exemples de répartition des points de la courbe en selon leurs valeurs pour des images générées avec l'algorithme de Machine Learning.

Source : Réalisation personnelle

Les quatre histogrammes ci-dessus confirment la concentration des points sur valeurs 0 et 255. Cela est principalement dû à la grande dominance du blanc (code RVB = 255,255,255) et du noir (code RVB = 0,0,0) dans les images. Cette concentration des points participe aux bons résultats obtenus puisqu'il y a peu de dispersion.

Conclusion :

Dans cette étude, on a pu comparer deux méthodes pour calculer le potentiel solaire. Une Technique de simulation le Ray Tracing et un algorithme de Machine Learning. L'apprentissage automatique, méthode de plus en plus utilisée montre aussi une très bonne performance dans le calcul du potentiel solaire, presque 100 fois plus rapide que la simulation avec des résultats presque identiques à ceux du Ray Tracing. De plus, notre algorithme semble être plus adapté pour le calcul du potentiel solaire sur de grandes surfaces car son temps de calcul est indépendant de la complexité du milieu contrairement à la méthode du Ray Tracing.

Dans l'optique d'études futures sur l'utilisation d'algorithmes d'intelligence artificielle pour le calcul du potentiel solaire, il est pertinent de définir les limites de cette étude.

La première limite est intrinsèque à notre calcul du coefficient de corrélation de Pearson. On a vu que la majorité des points des images étaient concentrés sur deux valeurs proches de 0 et de 255. Cela aide à obtenir un bon coefficient de corrélation comme si le nuage de points était constitué de deux uniques points.

Bibliographie :

- MINISTERE DE LA TRANSITION ECOLOGIQUE (s. d.). Chiffres clés de l'énergie - Édition 2021. Disponible sur : <<https://www.statistiques.developpement-durable.gouv.fr/edition-numerique/chiffres-cles-energie-2021/6-bilan-energetique-de-la-france>>.
- JOSHI SIDDHARTH. (2021, 5 octobre). *High resolution global spatiotemporal assessment of rooftop solar photovoltaics potential for renewable electricity generation*. Nature. Disponible sur : <https://www.nature.com/articles/s41467-021-25720-2?error=cookies_not_supported&code=38293237-8dc2-40be-9fc1-494ab389a7c7>.
- WIKIPEDIA CONTRIBUTORS. (2022, 30 novembre). *Ray tracing*. Disponible sur : <https://fr.wikipedia.org/wiki/Ray_tracing>.
- RIS Philippe. *Les modèles d'illumination en lancer de rayon*. Thèse de doctorat : Université de Franche-comté. UFR des sciences et techniques. Disponible sur : <<http://ph.ris.free.fr/these/Ph.RIS-Chp2Illumination.pdf>>.
- MAIZIA MINDJID (2014). *Déterminer les zones d'ensoleillement d'un espace urbain*. Toasterintegrale. Disponible sur : <https://daereseau.univ-tours.fr/ressource/toasterwebsite/document_demarrer.html>.
- LAROUSE, Æ. (s. d.). *Intelligence artificielle - LAROUSSE*. Disponible sur : <https://www.larousse.fr/encyclopedie/divers/intelligence_artificielle/187257>.
- GUILLAUME LEMAITRE, G. L. et al (2022, août 5). *Nearest Neighbor Classification*. GitHub. Disponible sur : <<https://github.com/scikit-learn/scikit-learn/blob/f3f51f9b6/sklearn/neighbors/classification.py>> et <<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>>.
- IBM, *Qu'est-ce que l'algorithme des k plus proches voisins ?* | IBM. (s. d.). Disponible sur : <<https://www.ibm.com/ca-fr/topics/knn>>.
- VILLE DE PARIS. (2013, juillet). *Cadastre solaire paris*. Paris.fr. Disponible sur : <<https://capgeo.sig.paris.fr/Apps/CadastreSolaire/>>.

Annexe :

Résultat :

Annexe 1 : Tableau des temps et corrélation pour générer des images de potentielle solaire avec l'algorithme du PFE :

id_image	n° image	Durée (s)	a	b	PCC
0	0	1,889035	0,989541	-4,914280	0,963735
1	1	1,903994	1,002471	-5,714447	0,971912
2	2	1,938263	0,979084	-6,864390	0,947178
3	3	1,900185	0,975756	-7,615019	0,936122
4	4	1,659067	0,938542	-0,742233	0,917795
5	5	1,511003	0,972201	0,479482	0,967368
10	6	1,576466	0,974604	-5,501169	0,943987
11	7	1,488407	0,987995	-3,143313	0,968114
12	8	1,709451	0,997976	-3,771014	0,973362
13	9	1,686255	0,978131	-5,826553	0,948629
14	10	1,748832	0,943083	1,487328	0,948757
15	11	1,611784	0,987803	-2,593792	0,968931
20	12	1,703868	0,999897	-3,997769	0,981378
21	13	1,720795	1,003853	-7,618614	0,967139
22	14	1,768463	0,991065	-4,859643	0,973352
23	15	1,775931	0,990104	-9,199985	0,962126
24	16	1,682378	0,976346	-7,342833	0,952323
25	17	1,771388	0,994896	-5,545379	0,961419
30	18	1,754877	1,010367	-5,825579	0,990863
31	19	1,665105	0,991073	-3,167764	0,979609
32	20	1,797785	0,996160	-6,269063	0,967382
33	21	1,766168	0,983809	-7,957754	0,953505
34	22	1,790533	0,978620	-9,325680	0,953058
35	23	1,522291	0,970887	-9,014635	0,947576
40	24	1,709874	0,985999	-5,166763	0,971454
41	25	1,753124	0,996018	-5,064927	0,967669
42	26	1,860401	0,998371	-6,535676	0,967114
43	27	1,777053	0,942465	0,656496	0,941932
44	28	2,224330	1,012873	-7,546619	0,969842
45	29	1,657108	0,992269	-7,287200	0,948445
50	30	1,598812	1,003153	-5,823827	0,970838
51	31	1,596619	1,016443	-5,667701	0,980820
52	32	1,711605	1,015639	-9,948073	0,968474
53	33	1,672693	0,983542	-8,640674	0,952339
54	34	1,710392	0,962018	-1,784742	0,950135
55	35	1,861902	0,997515	-0,793908	0,975841
60	36	1,744212	1,069357	-9,651308	0,980893
61	37	1,745359	1,030694	-6,743280	0,972020

62	38	1,747118	1,054911	-12,144478	0,977405
63	39	1,731533	1,054751	-12,993079	0,967878
64	40	1,685775	1,025886	-4,493640	0,982634
65	41	1,698973	1,018896	-0,812133	0,964025
70	42	1,738891	1,095083	-11,069045	0,988951
71	43	1,671776	1,091874	-9,907114	0,987682
72	44	1,707618	1,051836	-10,157985	0,974816
73	45	1,678500	1,098928	-13,960888	0,974427
74	46	1,727220	1,073399	-11,842678	0,965492
75	47	1,734699	1,030573	-8,919499	0,936256
80	48	1,856611	1,110410	-7,998095	0,993363
81	49	1,763038	1,129136	-10,304276	0,991413
82	50	1,723000	1,063269	-9,739425	0,987523
83	51	1,738625	1,139432	-12,257531	0,975731
84	52	1,698739	1,119825	-9,725572	0,981006
85	53	1,812224	1,071728	-9,057173	0,953342
90	54	1,684977	1,158017	-11,623079	0,982760
91	55	1,836717	1,176606	-14,016036	0,995457
92	56	1,615152	1,153617	-9,312319	0,994002
93	57	1,766928	1,179628	-12,356695	0,988612
94	58	1,718858	1,155815	-10,284313	0,987574
95	59	1,745201	1,136959	-10,745743	0,982952
100	60	1,672616	1,164584	-10,554081	0,983821
101	61	1,699360	1,206542	-13,408112	0,989507
102	62	1,773373	1,187246	-11,891696	0,987815
103	63	1,701913	1,205549	-14,895622	0,966951
104	64	1,655480	1,171633	-8,352150	0,932166
105	65	1,684920	1,188215	-12,939110	0,973830
2000	66	0,952606	1,161121	-10,539026	0,987526
2001	67	0,921028	1,120125	-3,872394	0,953247
2002	68	0,848750	1,192645	-13,559639	0,986708
2003	69	0,847268	1,167724	-10,677620	0,986134
2004	70	0,836116	1,174101	-12,099219	0,995064
2005	71	0,820652	1,178648	-13,428272	0,993351
2010	72	0,845269	1,080403	-9,904498	0,956486
2011	73	0,868293	1,113253	-8,651912	0,970632
2012	74	0,845142	1,112159	-7,448169	0,985180
2013	75	0,865795	1,122819	-11,941323	0,984082
2014	76	0,847326	1,102851	-11,049961	0,989884
2015	77	0,855810	1,121520	-7,681753	0,994620
2020	78	0,866651	1,048018	-10,959844	0,948722
2021	79	0,838886	1,069821	-10,743112	0,947387
2022	80	0,826426	1,085194	-12,128072	0,977504
2023	81	0,835483	1,089728	-11,208237	0,965907
2024	82	0,877444	1,082898	-11,848192	0,986291

2025	83	0,826005	1,096244	-9,137966	0,994037
2030	84	0,848659	1,001165	-3,370879	0,947147
2031	85	0,847654	1,011719	-0,228921	0,968163
2032	86	0,849172	1,048656	-13,536621	0,968110
2033	87	0,868846	1,070447	-12,706784	0,976041
2034	88	0,854989	1,049935	-11,241291	0,974418
2035	89	0,841058	1,054607	-8,893431	0,970407
2040	90	0,828705	1,003956	-3,177990	0,970517
2041	91	0,854486	0,971977	3,572809	0,956720
2042	92	0,888303	1,009332	-11,946723	0,957453
2043	93	0,845412	1,007974	-9,102648	0,964352
2044	94	0,850404	1,006329	-7,374484	0,967476
2045	95	0,849652	1,031257	-7,316657	0,974405
2050	96	0,870494	1,000265	-5,786100	0,965709
2051	97	0,863136	0,985004	-3,734639	0,953761
2052	98	0,878198	0,991444	-7,756469	0,954138
2053	99	0,827051	0,982023	-2,286404	0,967370
2054	100	0,854377	0,990511	-3,839244	0,964097
2055	101	0,864399	0,996589	-6,950744	0,965188
2060	102	0,844830	0,985124	-6,666535	0,950811
2061	103	0,835392	0,967506	-8,995858	0,954198
2062	104	0,836762	0,979087	-8,026767	0,955357
2063	105	0,868298	0,989677	-7,618492	0,964234
2064	106	0,878693	0,994942	-4,868412	0,971721
2065	107	0,855967	1,007356	-3,761969	0,992300
2070	108	0,854086	0,998883	-6,247939	0,961925
2071	109	0,872003	0,981166	-7,454411	0,956290
2072	110	0,880885	0,982306	-9,457353	0,953125
2073	111	0,817183	0,990314	-7,487031	0,962454
2074	112	0,846539	0,994885	-4,820546	0,971901
2075	113	0,849222	1,014812	-9,565768	0,976050
2080	114	0,875777	0,996577	-4,355191	0,978259
2081	115	0,848060	0,996323	-1,956792	0,980824
2082	116	0,869135	0,952813	-2,381664	0,947642
2083	117	0,877101	1,000049	-6,224387	0,969491
2084	118	0,996241	0,988720	-2,813896	0,971942
2085	119	0,920669	0,974829	-4,554562	0,948930
2090	120	0,947545	0,977469	-0,694036	0,970120
2091	121	0,844807	0,956217	0,759986	0,948644
2092	122	1,080392	0,947599	-4,892772	0,915578
2093	123	0,856036	0,989284	-8,454173	0,950162
2094	124	0,859124	0,989627	-4,813348	0,960301
2095	125	0,986098	1,007112	-5,211505	0,982271
3000	126	0,966216	1,170987	-13,249868	0,985732
3001	127	0,856081	1,081017	-11,447312	0,980326

3002	128	0,856673	1,016290	-11,337954	0,955235
3003	129	0,858142	0,997084	-8,219682	0,956260
3004	130	0,837452	0,981633	-6,457121	0,946684
3005	131	0,844272	0,982270	-7,782846	0,935962
3010	132	0,892483	1,139492	-11,643852	0,954615
3011	133	0,826245	1,069859	-11,018978	0,959831
3012	134	0,846547	1,006723	-11,495784	0,949778
3013	135	0,853970	0,966686	-8,188917	0,932537
3014	136	0,907087	0,971150	-8,802001	0,929520
3015	137	0,868567	0,951682	-4,722101	0,898792
3020	138	0,847004	1,121288	-9,816476	0,959214
3021	139	0,836421	1,005761	-6,087273	0,918751
3022	140	0,827544	0,948119	1,889303	0,912674
3023	141	0,856111	0,954604	-5,386423	0,918599
3024	142	0,904209	0,982075	-6,737475	0,940589
3025	143	0,921471	0,957251	-0,150938	0,929625
4000	144	1,288098	0,942086	-23,465778	0,945087
4001	145	1,100890	0,954795	-16,525655	0,971584
4002	146	1,504471	0,939115	-19,494682	0,923525
4003	147	1,482220	0,970958	-30,581194	0,952246
4004	148	1,018545	0,910538	-11,395284	0,957588
4005	149	1,311682	0,934979	-16,805914	0,980705
4010	150	0,955785	0,927237	-18,443409	0,970814
4011	151	0,925338	0,977968	-25,621841	0,965714
4012	152	0,892537	0,977297	-20,135332	0,978060
4013	153	0,905357	0,953941	-22,374002	0,972956
4014	154	0,975787	0,958816	-15,268697	0,969125
4015	155	0,884914	0,936283	-19,081886	0,980615
4020	156	0,925727	0,901174	-20,555410	0,915998
4021	157	0,871125	0,914162	-20,795098	0,933895
4022	158	0,849100	0,985705	-22,004662	0,977141
4023	159	0,942944	0,921799	-14,870320	0,969797
4024	160	0,861738	0,970446	-20,932889	0,968378
4025	161	1,059361	0,975439	-24,621549	0,970925
4030	162	0,895712	0,924536	-24,929514	0,911860
4031	163	0,918823	0,909423	-24,563811	0,911582
4032	164	0,947428	0,936066	-21,317302	0,933439
4033	165	0,849086	0,976658	-17,133867	0,968865
4034	166	1,017443	0,967593	-16,770151	0,966262
4035	167	0,941891	0,994379	-25,289657	0,979012
4040	168	0,906869	0,924624	-21,807606	0,924844
4041	169	2,135181	0,959214	-32,296993	0,936067
4042	170	2,173483	1,062623	-45,050846	0,966499
4043	171	0,949163	0,976990	-21,877436	0,958920
4044	172	0,895875	0,990463	-19,867100	0,972867

4045	173	0,999866	1,028011	-35,563329	0,961920
4050	174	0,898453	0,916481	-21,761177	0,922529
4051	175	2,006229	0,993574	-36,858360	0,936845
4052	176	2,207520	1,013678	-40,803501	0,950711
4053	177	0,939996	0,945068	-21,754138	0,953881
4054	178	0,901815	0,981006	-26,394600	0,957202
4055	179	1,038244	1,003531	-32,956377	0,956694
4060	180	0,868006	0,904645	-25,266274	0,911438
4061	181	0,928401	0,941728	-25,157305	0,926422
4062	182	0,996433	0,981495	-30,819852	0,945037
4063	183	1,127730	0,948737	-26,074313	0,949139
4064	184	1,180284	0,983412	-32,718463	0,943072
4065	185	1,061111	1,034053	-43,895626	0,944900
4070	186	0,862154	0,913041	-24,516023	0,907550
4071	187	0,853348	0,932536	-21,800270	0,939548
4072	188	0,855399	0,981043	-25,576026	0,961569
4073	189	0,897102	0,960955	-27,751671	0,942966
4074	190	1,082209	0,976473	-28,743267	0,940696
4075	191	1,104795	1,030502	-40,162144	0,958390
4080	192	0,917940	0,940872	-29,553690	0,920404
4081	193	0,908872	0,978543	-31,966207	0,933007
4082	194	0,947403	0,956013	-29,369414	0,932090
4083	195	0,897635	0,962362	-25,874027	0,945531
4084	196	1,010726	0,987767	-31,788333	0,951115
4085	197	0,990157	0,930411	-33,709834	0,918977
4090	198	1,309970	0,987126	-41,104100	0,926477
4091	199	1,350660	0,993073	-37,140096	0,933101
4092	200	1,110781	1,012479	-37,507105	0,949076
4093	201	0,886214	0,973337	-32,672781	0,929087
4094	202	1,090345	1,004225	-40,025025	0,935029
4095	203	0,930703	0,994701	-38,161168	0,928903
40100	204	1,698779	1,046621	-48,233454	0,950392
40101	205	1,758436	1,007074	-46,091631	0,922204
40102	206	1,605461	0,908725	-33,985430	0,886974
40103	207	0,872293	0,883985	-27,324831	0,889238
40104	208	1,086643	0,890633	-26,917398	0,898279
40105	209	1,284954	0,998338	-41,101289	0,929860
40110	210	1,730710	1,005995	-44,365273	0,927037
40111	211	1,534874	0,945367	-38,813913	0,896034
40112	212	1,019755	0,897181	-31,456492	0,890356
40113	213	0,868300	0,937158	-32,401872	0,916197
40114	214	0,900558	0,970474	-36,208156	0,929976
40115	215	0,898023	0,964034	-36,106683	0,918057
5000	216	1,868787	0,982525	2,402845	0,976194
5001	217	2,383090	0,954479	4,586560	0,972265

5002	218	3,028848	0,993415	2,171953	0,971477
5003	219	2,247070	0,975488	3,951587	0,983441
5004	220	1,657399	0,898309	3,958756	0,902845
5005	221	1,608531	0,939595	1,562193	0,919679
5006	222	1,453218	0,989312	0,669856	0,968050
5010	223	1,697155	1,012046	-0,433113	0,957086
5011	224	1,171664	1,015516	-1,003263	0,973502
5012	225	1,355800	0,951901	0,015885	0,981472
5013	226	1,874176	0,968964	0,204283	0,964725
5014	227	1,499502	0,926967	0,806635	0,941317
5015	228	0,940334	1,010440	-5,713412	0,971718
5016	229	0,906405	1,020156	-6,514698	0,961344
5020	230	1,431393	0,961812	0,159860	0,952146
5021	231	0,999544	1,027451	-9,660869	0,988590
5022	232	1,609175	0,949926	-5,758930	0,982511
5023	233	1,415986	0,909260	-0,815028	0,961327
5024	234	0,931634	1,002019	-8,855346	0,965919
5025	235	0,860573	0,982604	-6,237257	0,978822
5026	236	0,886601	1,015957	-9,526806	0,973804
5030	237	1,250067	0,936451	-3,370072	0,946757
5031	238	1,033744	1,011958	-8,513212	0,984837
5032	239	0,977730	0,929439	-4,256786	0,972587
5033	240	1,002353	0,933208	-5,612038	0,966867
5034	241	0,986939	1,021251	-9,426641	0,979336
5035	242	1,049098	0,973249	-8,327988	0,977105
5036	243	1,125650	1,001861	-11,042583	0,970238
5040	244	2,317090	0,956556	-3,481631	0,962922
5041	245	1,194862	0,982680	-10,376398	0,967911
5042	246	1,177387	0,964082	-9,423382	0,967365
5043	247	1,064250	0,970173	-8,851362	0,980588
5044	248	0,850889	1,011647	-11,873169	0,985359
5045	249	0,997675	0,985123	-10,943875	0,986116
5046	250	0,915801	0,988094	-14,316758	0,980678
5050	251	1,344710	0,896541	-3,850033	0,949519
5051	252	0,918398	0,942296	-8,951183	0,966849
5052	253	0,991114	0,946845	-10,469043	0,955382
5053	254	0,872043	0,984831	-12,657005	0,973476
5054	255	1,006872	0,990589	-11,719681	0,973422
5055	256	0,993644	0,970303	-9,170514	0,979705
5056	257	1,022765	0,962971	-11,480447	0,978065
5060	258	1,616005	0,884083	-3,344291	0,943308
5061	259	0,948837	0,938768	-11,822617	0,960176
5062	260	0,976134	0,914921	-8,846798	0,950769
5063	261	0,858505	0,957673	-11,856176	0,970760
5064	262	0,854467	0,949717	-10,533570	0,974573

5065	263	0,871688	0,977553	-10,270654	0,979257
5066	264	0,887056	0,970833	-11,034460	0,969316
5070	265	1,169555	0,881480	-6,984764	0,938690
5071	266	1,074045	0,917531	-10,957697	0,949391
5072	267	0,986780	0,941476	-10,623885	0,968258
5073	268	1,027703	0,968430	-10,090391	0,977955
5074	269	1,022424	0,945036	-12,115501	0,972547
5075	270	0,929693	0,950573	-12,472954	0,965267
5076	271	0,922821	0,962661	-16,615055	0,968364
5080	272	1,701213	0,891968	-5,565465	0,946883
5081	273	0,917512	0,899871	-10,843910	0,945441
5082	274	0,964492	0,906894	-11,641598	0,940362
5083	275	1,011847	0,955955	-16,172462	0,963168
5084	276	0,911119	0,938813	-13,758070	0,970662
5085	277	1,218615	0,952227	-15,565827	0,971423
5086	278	1,299399	0,967765	-16,918702	0,973065
5090	279	1,170542	0,839034	-6,789785	0,898321
5091	280	0,986395	0,907737	-12,302984	0,943090
5092	281	1,086307	0,900905	-13,100155	0,935371
5093	282	0,948416	0,813051	-8,289258	0,882694
5094	283	1,303890	0,889021	-9,646118	0,935632
5095	284	1,108145	0,927548	-13,944030	0,954773
5096	285	1,902016	0,718142	3,543608	0,776340
50100	286	1,172750	0,860531	-10,613516	0,916980
50101	287	0,966584	0,860767	-12,252652	0,911560
50102	288	0,917928	0,916553	-16,862765	0,940719
50103	289	1,014540	0,870660	-13,326855	0,916446
50104	290	0,895090	0,885871	-13,512235	0,927781
50105	291	0,922330	0,930069	-17,831991	0,952371
50106	292	2,218008	0,954272	-13,933412	0,966313
50110	293	1,239600	0,839095	-1,688090	0,895100
50111	294	1,130749	0,876540	-12,309767	0,933690
50112	295	0,948377	0,885345	-15,547258	0,932019
50113	296	0,990367	0,886445	-15,996676	0,925445
50114	297	1,065134	0,818641	-11,286455	0,878245
50115	298	0,996838	1,020247	-23,856354	0,981729
50116	299	2,752631	0,965386	-14,504473	0,944623
50120	300	1,082201	-4,582649	511,742565	-0,638556
50121	301	1,604071	0,865815	-4,113507	0,940671
50122	302	1,403424	0,863208	-8,629953	0,915709
50123	303	1,238988	0,894925	-13,486225	0,934391
50124	304	2,106022	0,878735	-8,905266	0,917298
50125	305	1,987628	0,882102	-9,498650	0,949782
50126	306	2,387772	0,718832	9,632489	0,317886
	Moyenne	1,225718	0,971222	-10,964804	0,948860

	Minimal	0,817183	-4,582649	-48,233454	-0,638556
	Maximal	3,028848	1,206542	511,742565	0,995457
	Médiane	0,999866	0,982525	-10,157985	0,961925

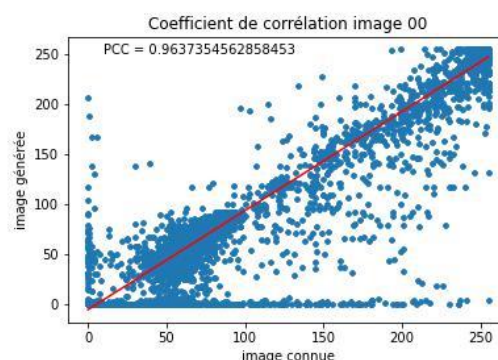
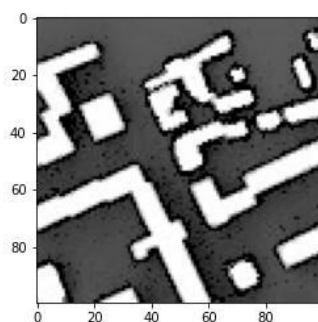
Annexe 2 : Tableau des temps pour générer des images de 100 pixels par 100 pixels avec la technique de Ray Tracing :

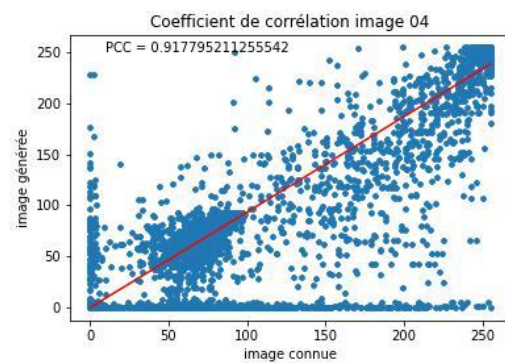
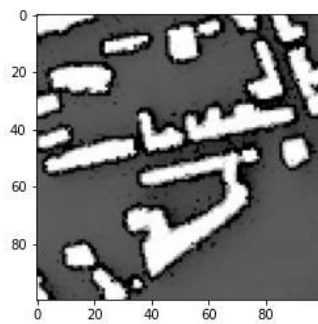
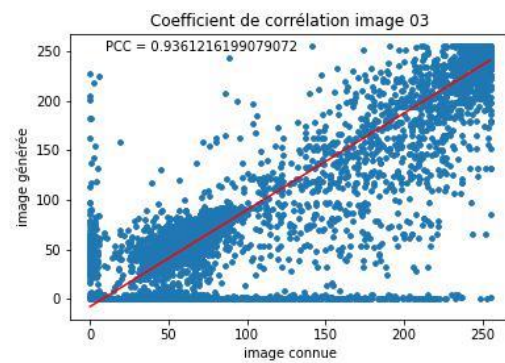
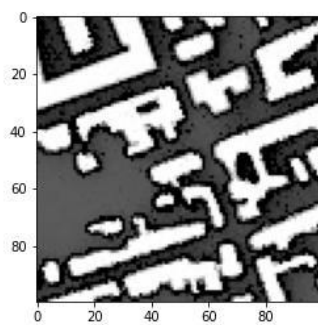
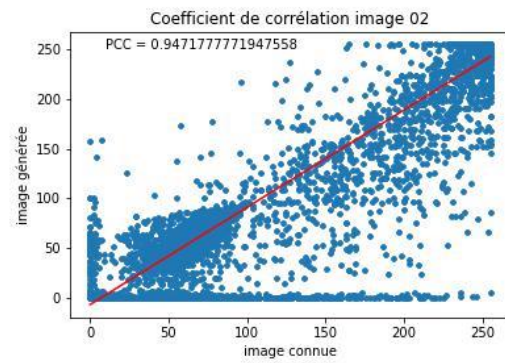
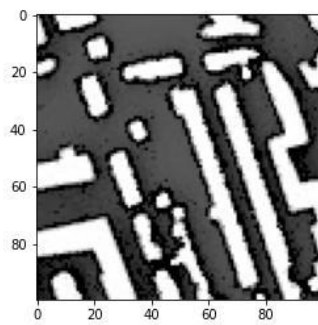
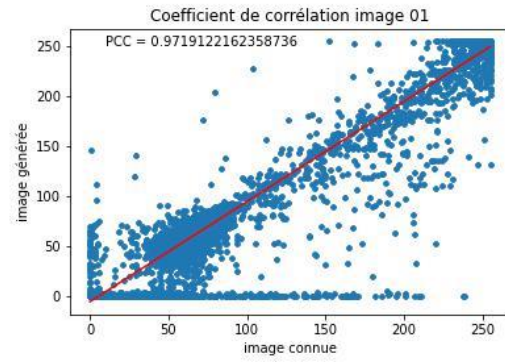
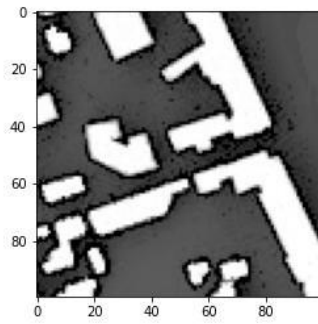
Image	durée (s)
1	204
2	134
3	117
4	177
5	83
6	115
7	151
8	55
9	85
10	115
11	72
12	113
13	138
14	111
15	109
Moyenne	118,6
Minimal	55
Maximal	204
Médiane	115

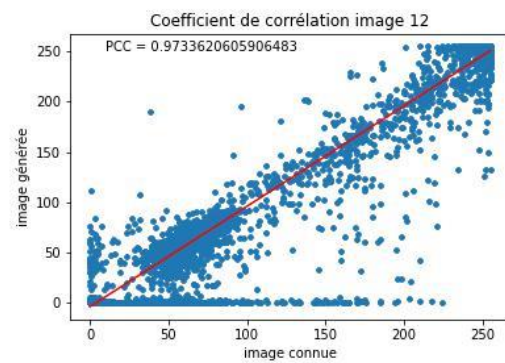
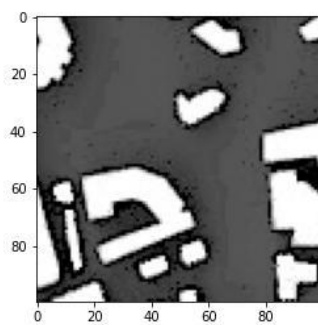
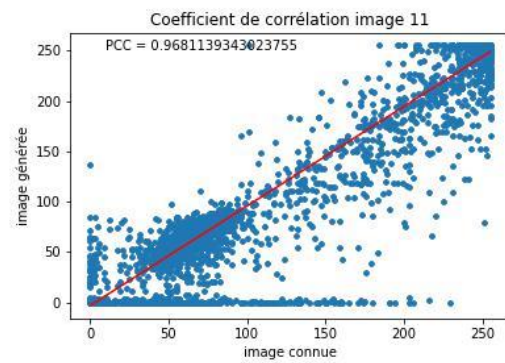
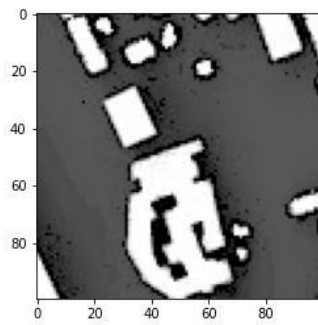
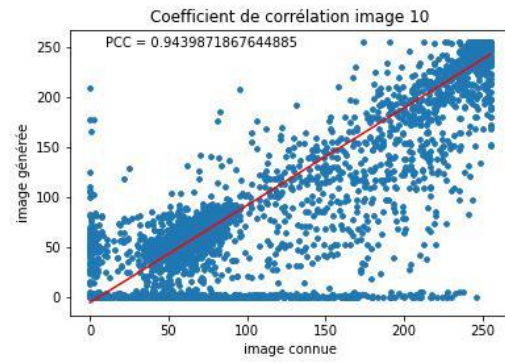
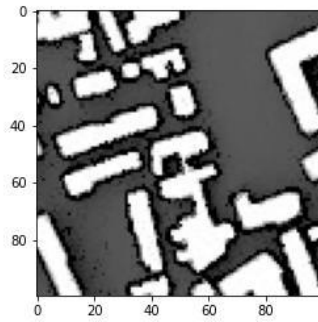
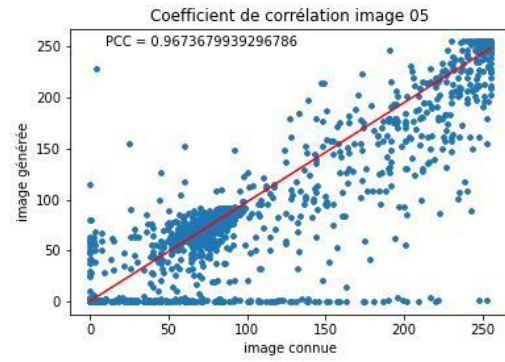
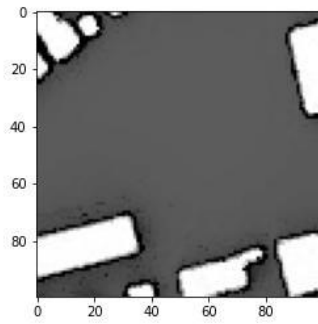
Echantillon d'images générées avec l'algorithme avec leur nuage de points, droite et coefficient de corrélation correspondant :

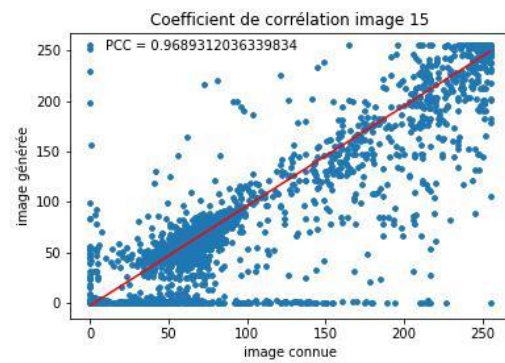
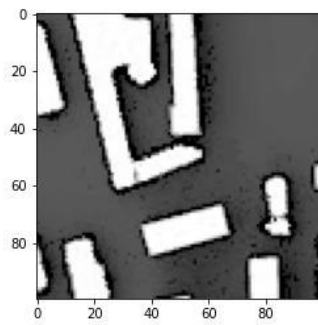
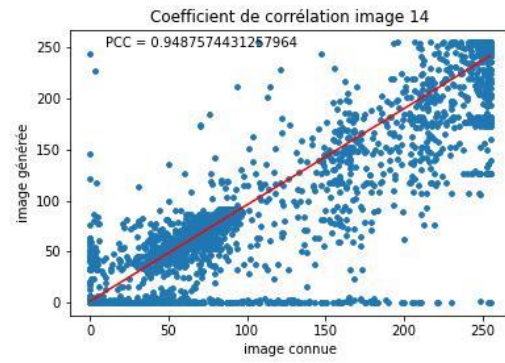
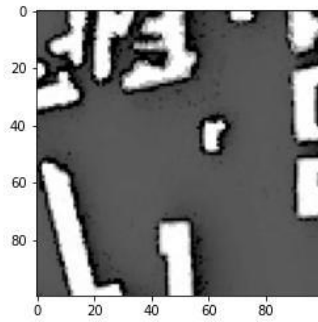
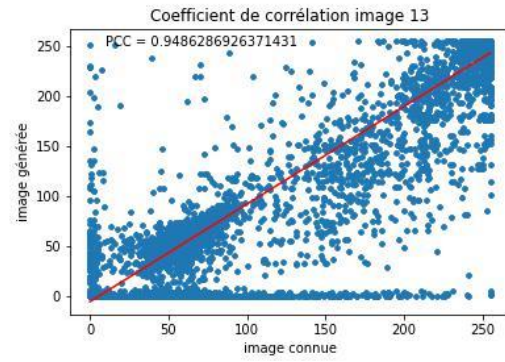
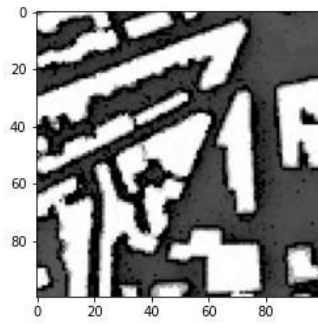
Annexe 3 : Echantillon d'images des 5 séries d'image de potentielle solaire réalisées avec notre algorithme ainsi que le nuage de points, la droite et le coefficient de corrélation correspondant.

Série 1 :

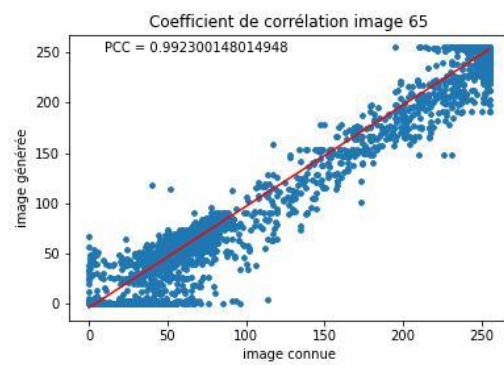
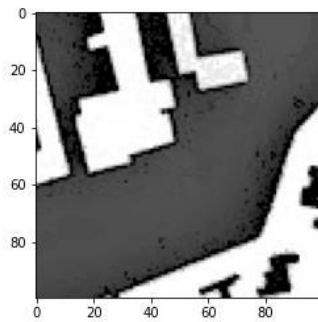


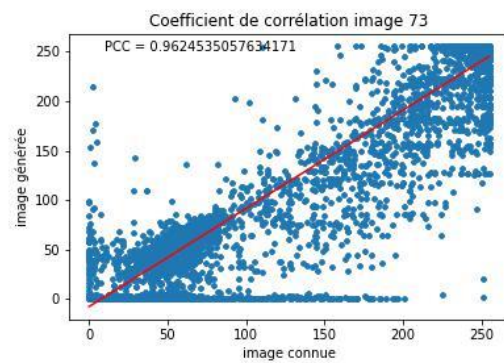
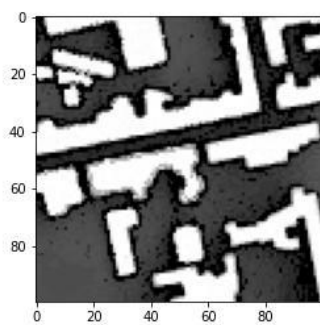
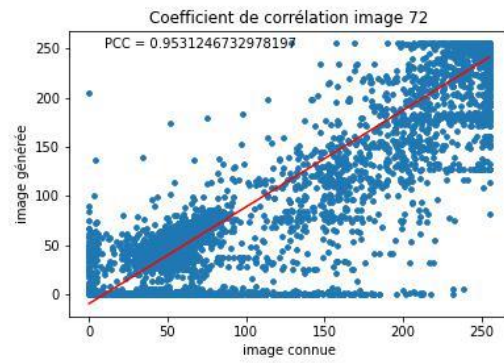
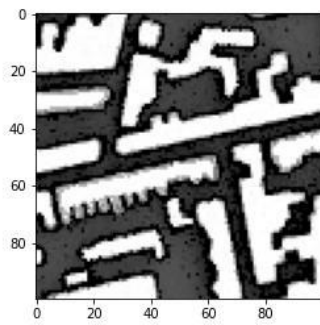
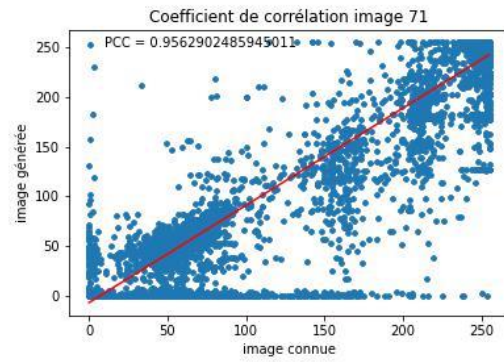
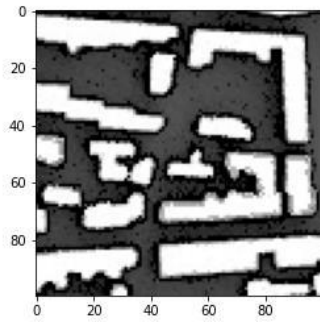
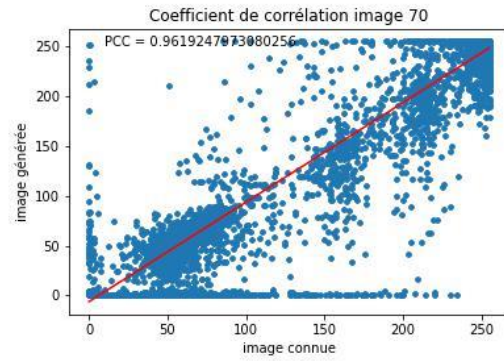
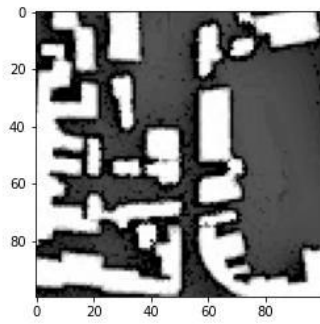


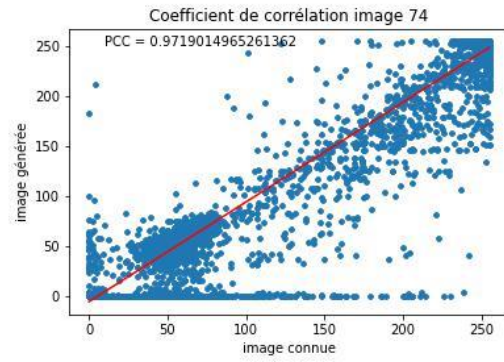
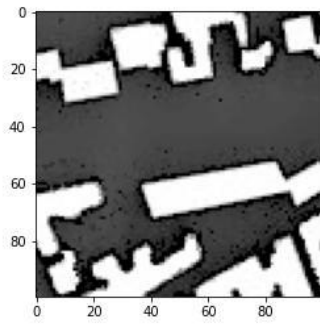




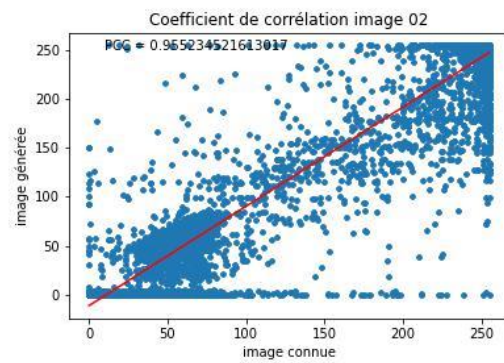
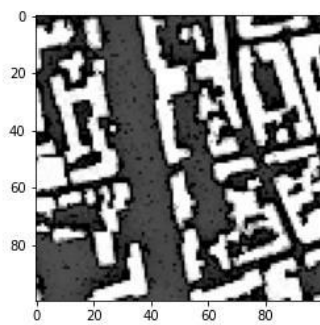
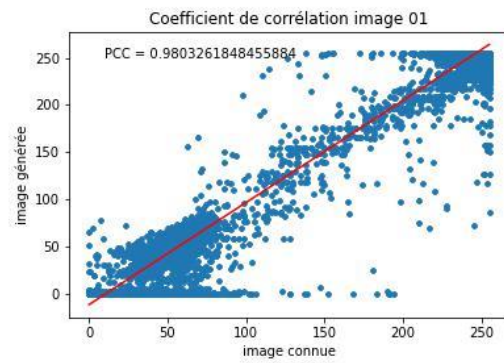
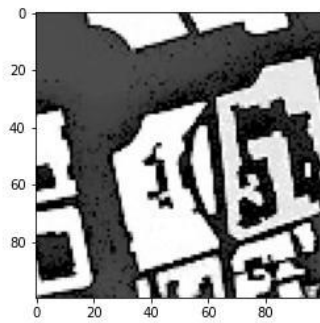
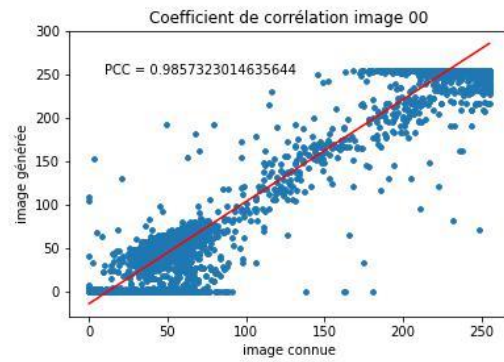
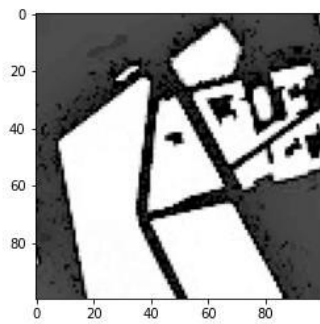
Série 2 :

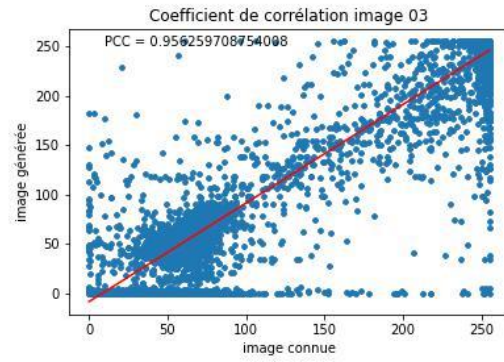
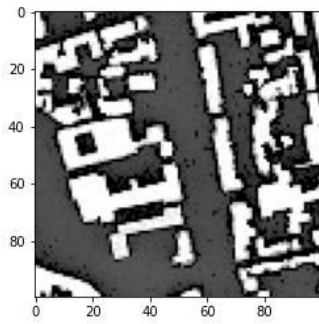




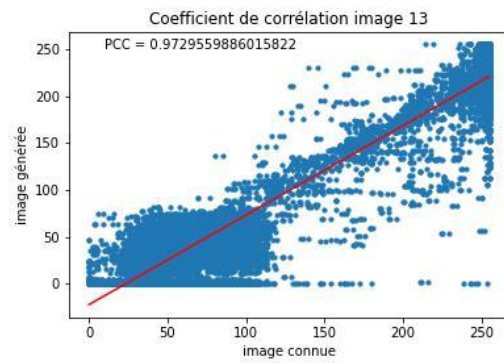
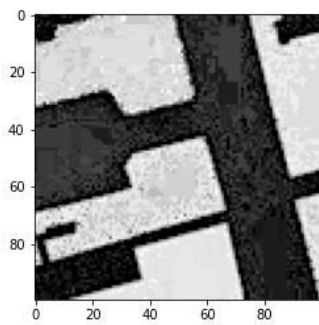
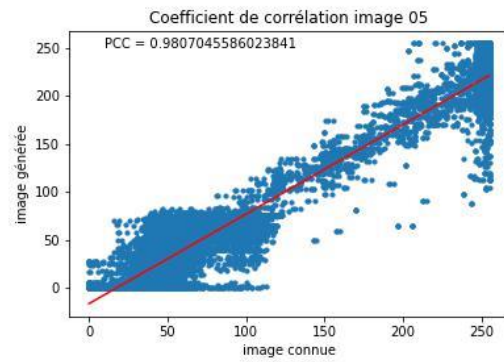
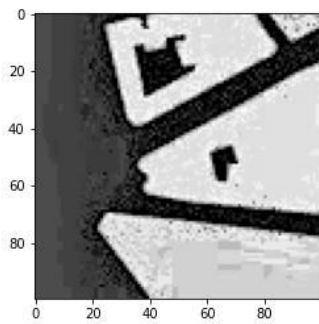
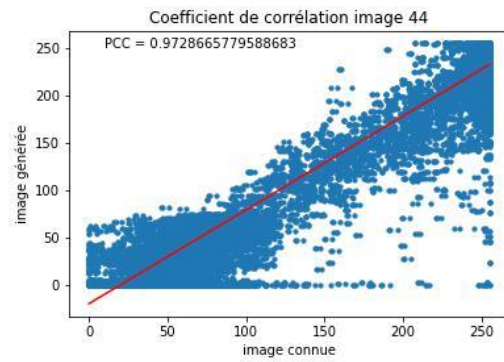
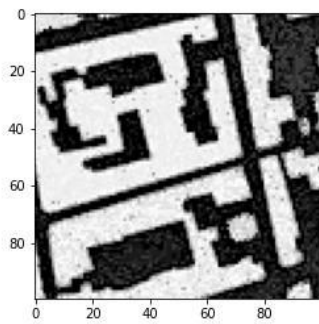


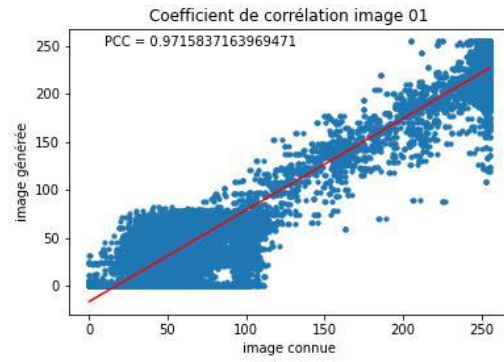
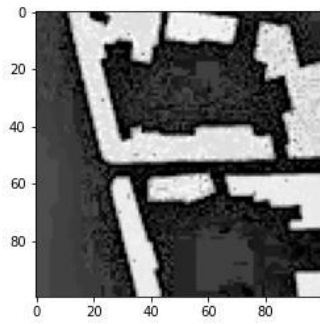
Série 3 :



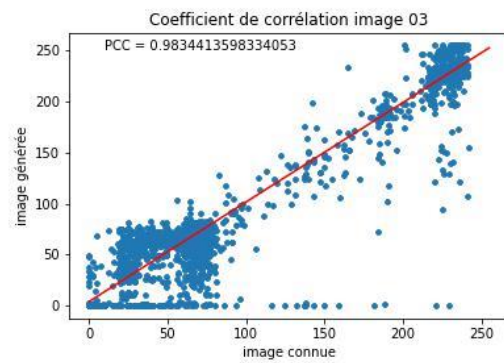
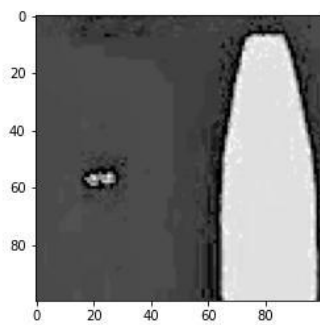
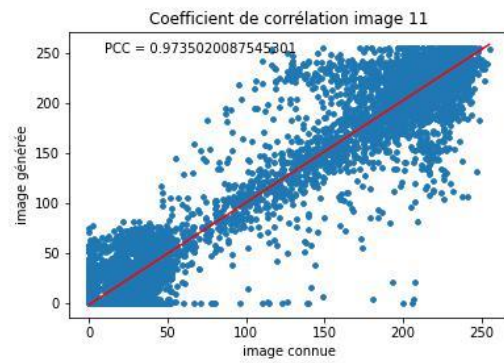
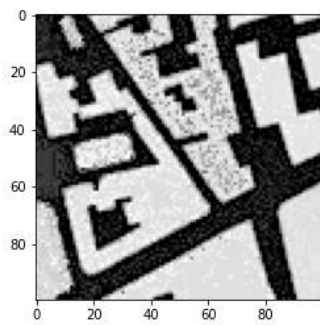
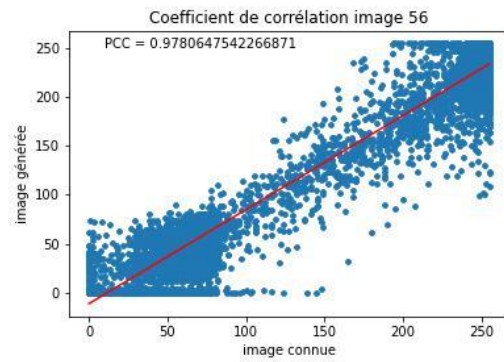
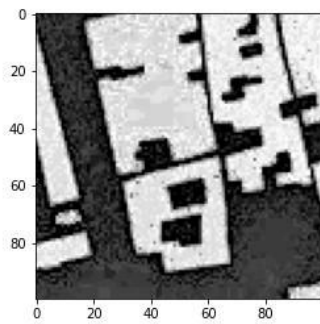


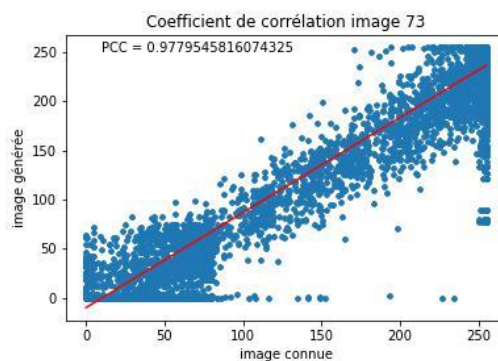
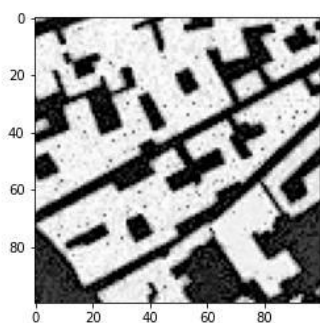
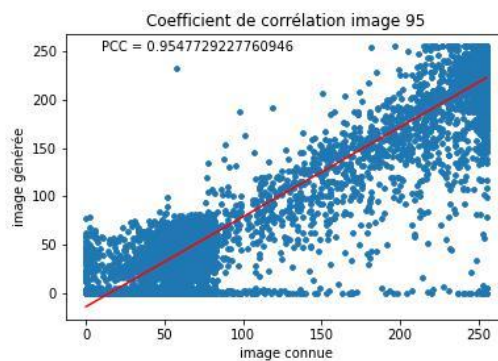
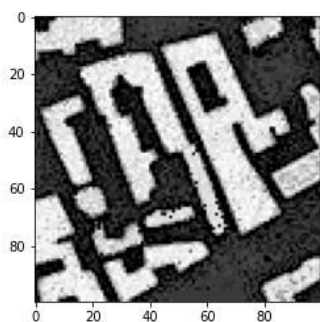
Série 4 :





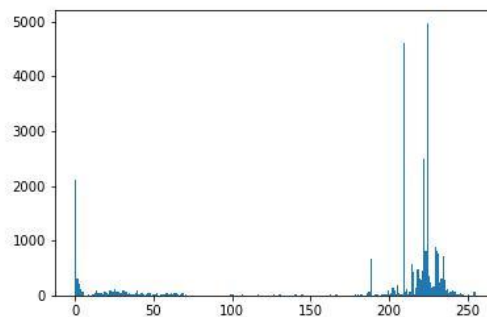
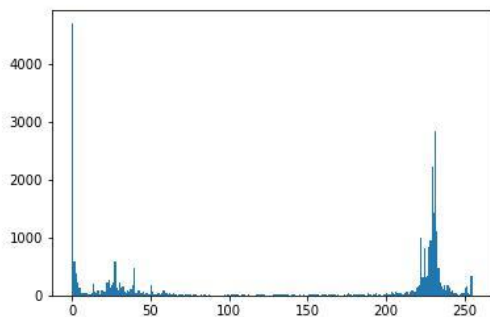
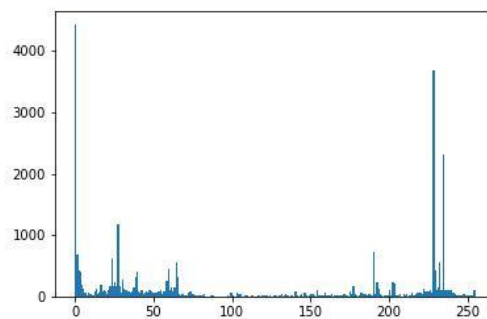
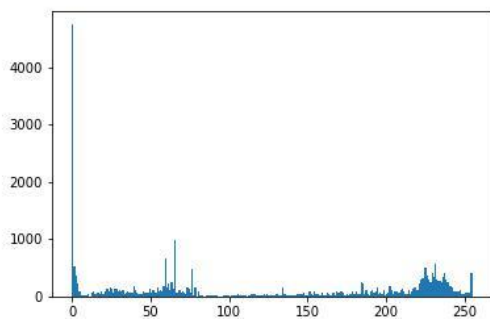
Série 5 :

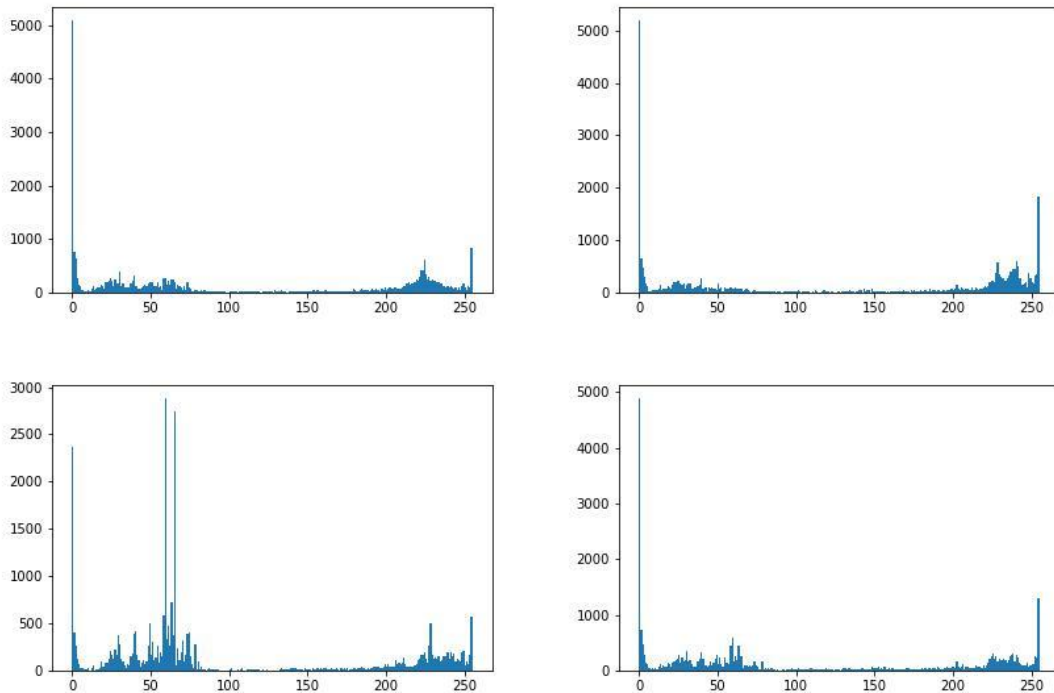




Echantillon de la répartition des points selon leur valeur :

Annexe 4 : échantillon de 8 histogrammes de la répartition des points selon leur valeur parmi les 307 images générées





Construction de l'algorithme

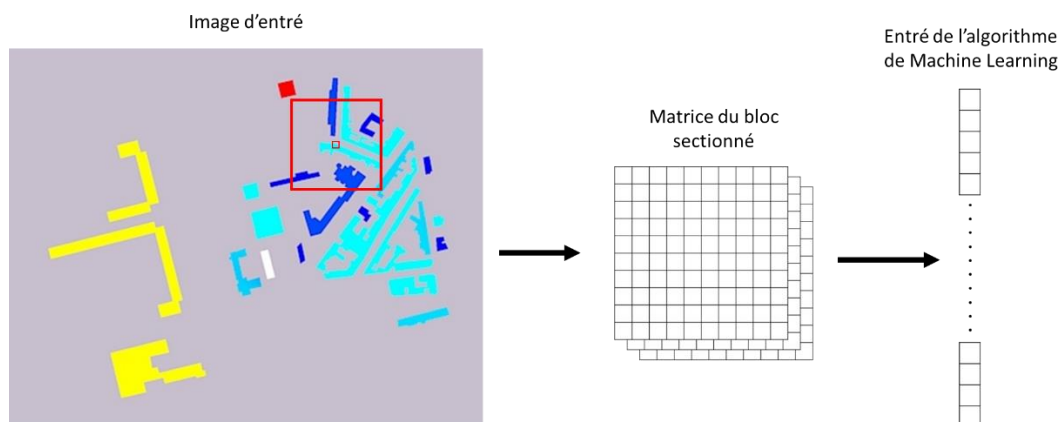
Annexe 5 : Détail de la construction de l'algorithme.

De l'image aux données en entrée de l'IA,

L'entrée de l'algorithme est une image. Toutefois, l'algorithme de Machine Learning ne prend en entrée que des listes de valeurs.

Une image est une matrice de trois dimensions avec des valeurs comprise entre 0 et 255. On sélectionne un bloc de p pixels autour du pixel à déterminer. « p » est une valeur variable qui dépend de l'échelle et de la taille de l'image.

Une fois le bloc sélectionné, on obtient une matrice de dimension $p \times p \times 3$. Cette matrice est réduite à un unique vecteur. Ce vecteur, est la donnée d'entrée pour l'algorithme de Machine Learning.



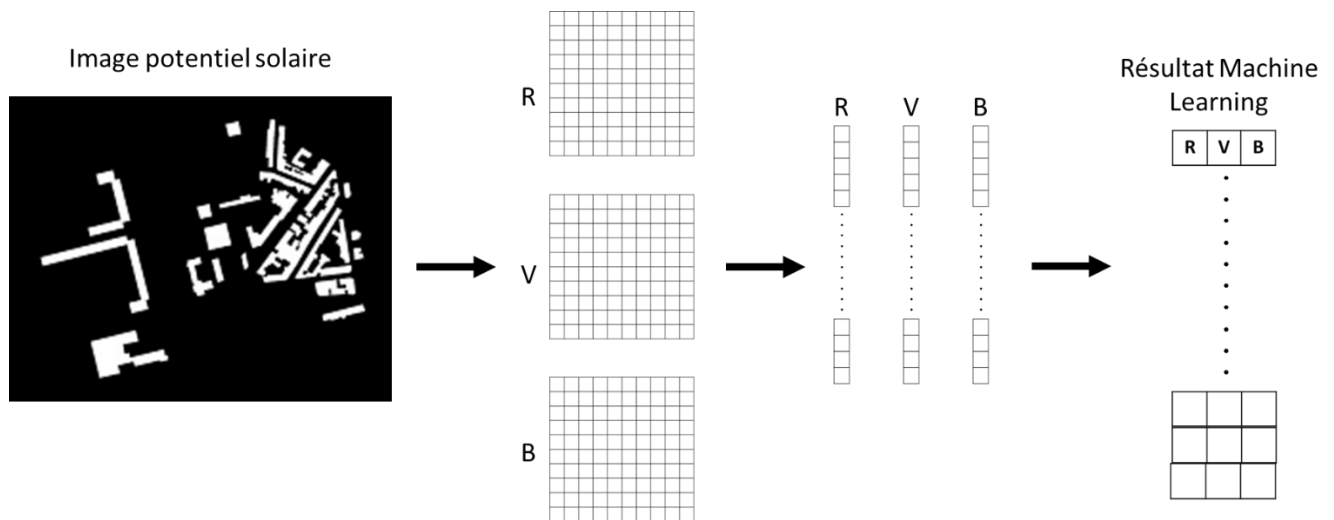
Donnée de sortie de l'algorithme de Machine Learning,

L'algorithme de Machine Learning nous donne comme résultat un vecteur de 3 valeurs. Ce vecteur correspond à un pixel dont la couleur représente le potentiel solaire.

Dans la phase d'entraînement de l'algorithme, il est nécessaire de créer une donnée de sortie.

Pour cela, on part d'image de potentiel solaire obtenu avec SketchUp et Toaster intégrale. On sépare la matrice de l'image en trois matrices (ces matrices correspondent aux valeurs Rouges, Vert, bleu de chaque pixel). On passe les trois matrices en colonnes, puis on les assemble. On obtient une matrice de 3 colonnes et autant de lignes que de pixels.

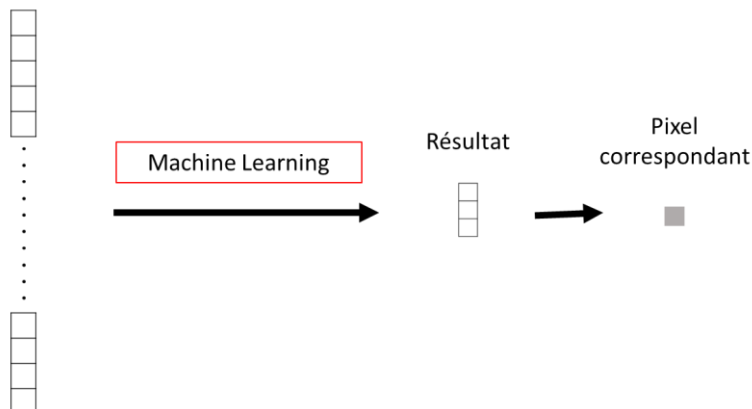
Chaque ligne correspond au code RVB d'un pixel.



Machine Learning :

L'algorithme de Machine Learning prend en entrée un vecteur de valeur correspondant à un bloc de « p » pixel autour du pixel recherché, et renvoie un vecteur de trois valeurs qui est un pixel.

Entrée de l'algorithme
de Machine Learning

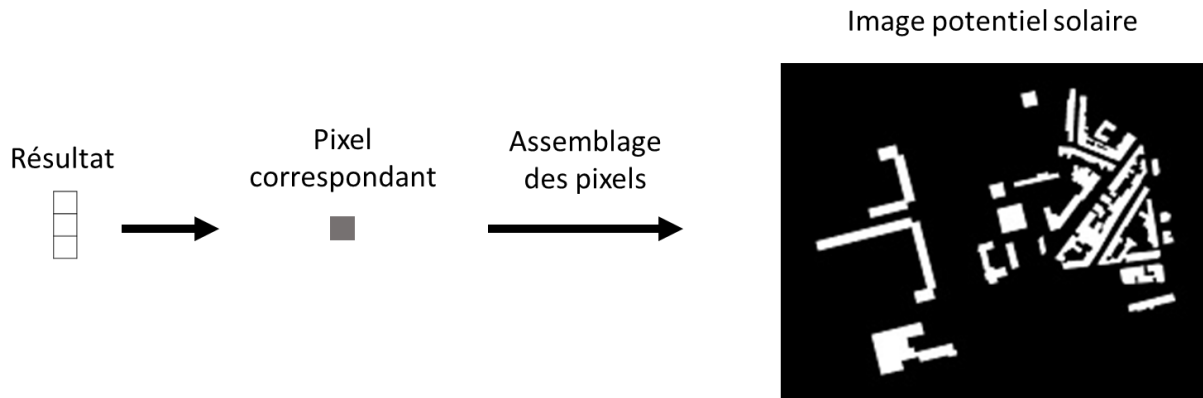


L'algorithme de Machine Learning est issu de la bibliothèque Scikit-Learn. Scikit-Learn est une bibliothèque open source qui propose des algorithmes d'apprentissage automatique.

Donnée de sortie, Résultat :

L'algorithme de Machine Learning nous donne une matrice de dimension 3 x nombre de pixels. On redimensionne la matrice selon les dimensions de l'image d'entrée et on obtient l'image du potentiel solaire.

Pour visualiser l'image, on utilise le module matplotlib.



Code de l'algorithme :

Algorithme

Annexe 6 : code l'algorithme réaliser lors de ce PFE pour déterminer le potentiel solaire

#Algorithme de pixel vers pixel

```
def InitLinearLearn(X='image/teste_b.jpg',Y='image/teste_s.jpg'):
```

```
    # Importation des modules
```

```
    from sklearn.neighbors import KNeighborsClassifier
```

```
    import numpy as np
```

```
    import matplotlib.pyplot as plt
```

```
    # chargement de l'image d'entrainement (entrée)
```

```
    image_1=plt.imread(X)
```

```
    plt.imshow(image_1)
```

```
    image_1.shape
```

```
# traitement de l'image entrée
```

```
image_1_1=image_1[:, :, 0].ravel()  
print(image_1_1.shape)  
image_1_2=image_1[:, :, 1].ravel()  
image_1_3=image_1[:, :, 2].ravel()  
image_1f=np.vstack((image_1_1,image_1_2,image_1_3))  
image_1f=image_1f.transpose()  
image_1f.shape
```

```
# chargement de l'image d'entrainement (resultat)
```

```
image_2=plt.imread(Y)  
plt.imshow(image_2)  
image_2.shape
```

```
# traitement de l'image résultat
```

```
image_2_1=image_2[:, :, 0].ravel()  
print(image_2_1.shape)  
image_2_2=image_2[:, :, 1].ravel()  
print(image_2_2.shape)  
image_2_3=image_2[:, :, 2].ravel()  
print(image_2_3.shape)  
image_2f=np.vstack((image_2_1,image_2_2,image_2_3))  
image_2f=image_2f.transpose()  
image_2f.shape
```

```
# Entrainement du modèle
```

```

model=KNeighborsClassifier()
model=model.fit(image_1f,image_2f)

return model

def CalLinearLearn (image,model):

    # Importation des modules

    from sklearn.neighbors import KNeighborsClassifier
    import numpy as np
    import matplotlib.pyplot as plt

    # chargement de l'image à traiter

    image_3=plt.imread(image)
    plt.imshow(image_3)
    image_3.shape

    # traitement de l'image à traiter

    image_3_1=image_3[:, :,0].ravel()
    image_3_2=image_3[:, :,1].ravel()
    image_3_3=image_3[:, :,2].ravel()
    image_3f=np.vstack((image_3_1,image_3_2,image_3_3))
    image_3f=image_3f.transpose()
    image_3f.shape

    # calcul du potentiel solaire

```

```

image_r=model.predict(image_3f)
image_r=image_r.reshape((image_3.shape[0],image_3.shape[1],image_3.shape[2]))
plt.imshow(image_r)
#image_r.shape

return image_r #resultat image

#teste fonction
#model=InitLinearLearn()
#image=CallLinearLearn('image/teste_b.jpg',model)

#Algo avec cadrillage vers un pixel

def InitBlocLearn (X='image/teste_b.jpg',Y='image/teste_s.jpg'):

    p=10
    p2=int(p/2)
    p2

    # Importation des modules

    from sklearn.neighbors import KNeighborsClassifier
    import numpy as np
    import matplotlib.pyplot as plt

    # chargement de l'image d'entrainement (entrée)

    image_1=plt.imread(X)

```

```

plt.imshow(image_1)

#image_1.shape

# traitement de l'image entrée

    # découpage des blocs de taille pxp
image_1f=[]
for i in range (image_1.shape[1]-p):
    for j in range (image_1.shape[0]-p):
        image_1i=image_1[i:i+p,j:j+p,0:image_1.shape[2]].ravel()
        image_1f=np.concatenate((image_1f, image_1i), axis=0)
#image_1f.shape

#création de la matrice d'entrainement à partir des blocs de l'image d'entrainement

c=p*p*3
v=int(image_1f.shape[0]/c)
image_1f=image_1f.reshape((v,c))
#image_1f.shape
print("bloc")

# chargement de l'image d'entrainement (resultat)

image_2=plt.imread(Y)
plt.imshow(image_2)
#image_2.shape

# traitement de l'image résultat

image_2_1=image_2[p2:-p2,p2:-p2,0].ravel()
#print(image_2_1.shape)

```



```

image_2_2=image_2[p2:-p2,p2:-p2,1].ravel()
#print(image_2_2.shape)
image_2_3=image_2[p2:-p2,p2:-p2,2].ravel()
#print(image_2_3.shape)
image_2f=np.vstack((image_2_1,image_2_2,image_2_3))
image_2f=image_2f.transpose()
#image_2f.shape
print("sortie")
# Entrainement du modèle

model=KNeighborsClassifier()
model=model.fit(image_1f,image_2f)

return model

def CalBlocLearn (image,model) :

    p=10

    # Importation des modules

    from sklearn.neighbors import KNeighborsClassifier
    import numpy as np
    import matplotlib.pyplot as plt

    # chargement de l'image à traiter

    image_3=plt.imread(image)
    plt.imshow(image_3)
    #image_3.shape

```

```
# traitement de l'image à traiter
```

```
image_3f=[]
```

```
for i in range (image_3.shape[1]-p):
```

```
    for j in range (image_3.shape[0]-p):
```

```
        image_3i=image_3[i:i+p,j:j+p,0:image_3.shape[2]].ravel()
```

```
        image_3f=np.concatenate((image_3f, image_3i), axis=0)
```

```
#print(image_3f.shape)
```

```
c=p*p*3
```

```
v=int(image_3f.shape[0]/c)
```

```
image_3f=image_3f.reshape((v,c))
```

```
#image_3f.shape
```

```
# calcul du potentiel solaire
```

```
image_r=model.predict(image_3f)
```

```
image_r=image_r.reshape((image_3.shape[0]-p,image_3.shape[1]-p,image_3.shape[2]))
```

```
plt.imshow(image_r)
```

```
#image_r.shape
```

```
return image_r #resultat image
```

Simulation et Teste

Annexe 7 : Code pour tester l'algorithme

```
# Importation des modules
```

```
from sklearn.neighbors import KNeighborsClassifier
```

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.metrics import *
```

```
import time
```

```

from Sk_solar import IASolarPotential as asp

from scipy.stats import pearsonr

# Entrainement du modèle:

fin=time.time()
x='C:/Users/PC/Documents/PFE/image/input/B.jpg'
y='C:/Users/PC/Documents/PFE/image/connu/C.jpg'
model=asp.InitLinearLearn(x,y)
debut=time.time()
print("générer le model a pris :"+str(fin-debut)+" secondes")

# Utilisation de l'agorithme

for k in range (0,13):
    for i in range (0,7):
        plt.clf()
        # Générer une image de potentiel solaire
        chemin='C:/Users/PC/Documents/PFE/image/input/X5/'+str(k)+str(i)+'.jpg'
        debut=time.time()
        image=asp.CalLinearLearn(chemin,model)
        fin=time.time()
        plt.imshow(image)
        name_image='im_cs_50'+str(k)+str(i)
        # Sauvegarde de l'image
        plt.savefig('C:/Users/PC/Documents/PFE/image/generer/G5/'+name_image+'.jpg')
        plt.close()

```

```

# Récupération de l'image de Ray Tracing correspondante
Y='C:/Users/PC/Documents/PFE/image/connu/Y5/'+str(k)+str(i)+'.jpg'

imR=plt.imread(Y)
imR=imR.ravel()
imG=image.ravel()

# Calcul des coefficients de la droite de corrélation
D=np.polyfit(imR,imG,1)

# Calcul du coefficient de corrélation de Person
R2=pearsonr(imR,imG)

print("Générer l'image 50"+str(k)+str(i)+" a pris :"+str(fin-debut)+" secondes, les coefficients de
la droite linéaire de corrélation sont :"+str(D[0]),str(D[1])+" , le coefficients de correlation est
:"+str(R2[0]))

# génère le nuage de point avec en abscisse l'image de référence (image de Ray Tacing), et en
ordonnée l'image généré

plt.clf()
x=[0, 255]
y=[0+D[1], 255*D[0]+D[1]]
plt.scatter(imR,imG,s=10)
plt.plot(x,y,c='red')
plt.title('Coefficient de corrélation image '+str(k)+str(i))
plt.text(10,250,'PCC = '+str(R2[0]))
plt.xlabel('image connue')
plt.ylabel('image générée')

# sauvegarde du nuage de point avec la droite de corrélation et le coefficient de corrélation de
Person

plt.savefig('C:/Users/PC/Documents/PFE/image/generer/G5/PCC_'+str(k)+str(i)+'.jpg')
plt.close()

```

Méthode du Ray Tracing :

Annexe 8 : Protocole de la méthode de Ray Tracing utilisé pour calculer le potentiel solaire

Etape 1 : modélisation de l'espace urbain :

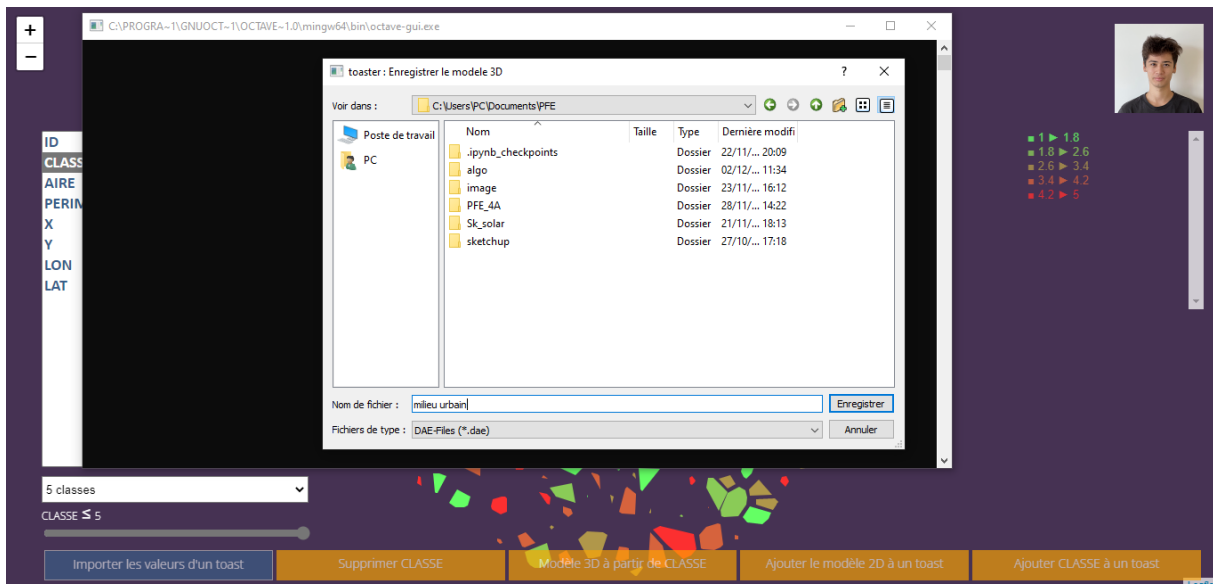


The screenshot shows the main menu of the Toasterdata application. It features a dark purple background with a sidebar on the left containing icons for different data types: a document with a function $y = f(x)$, a grid with numbers, a network diagram, a 2D map, and a 3D cube. The main area lists five options, each with a title and a brief description:

- Créer des données à partir d'un toast**: Enregistrer et modifier les données d'un toast avec toaster data. Cela vous permettra d'y voir plus clair en séparant soigneusement votre raisonnement des données auxquelles il s'applique.
- Créer des données numériques**: Créer ou importer, puis enregistrer et modifier un jeu de données correspondant au format du problème que vous cherchez à traiter.
- Créer un modèle 1D (polylignes)**: Créer ou importer une carte contenant un réseau (routes, réseau de distribution, etc.) afin de réaliser des calculs et de présenter des résultats cartographiques.
- Créer un modèle 2D (des polygones)**: Créer ou importer une carte contenant des polygones (bâtiments, occupation du sol, etc.) afin de réaliser des calculs et de présenter des résultats cartographiques.
- Créer un modèle 3D (des solides)**: Importer des modèles 3D (au format collada de Sketchup) afin de réaliser des modélisations en trois dimensions.

At the bottom of the main area, there are two orange buttons: "Créer ..." and "Importer ...".

Importer un plan urbain au format shapefile dans Toasterdata.

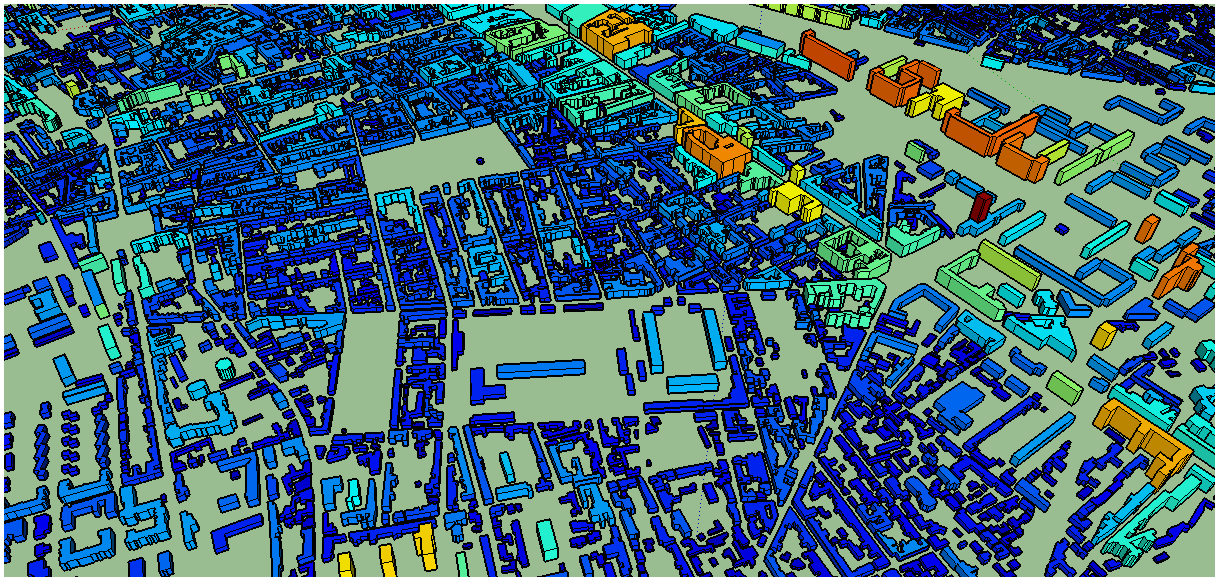


The screenshot shows the Toasterdata application with a file selection dialog open. The dialog is titled "toaster : Enregistrer le modèle 3D" and shows the contents of the folder "C:\Users\PC\Documents\PFEE". The files listed are:

Nom	Taille	Type	Dernière modif
.ipynb_checkpoints		Dossier	22/11/... 20:09
algo		Dossier	02/12/... 11:34
image		Dossier	23/11/... 16:12
PFE_4A		Dossier	28/11/... 14:22
Sk_solar		Dossier	21/11/... 18:13
sketchup		Dossier	27/10/... 17:18

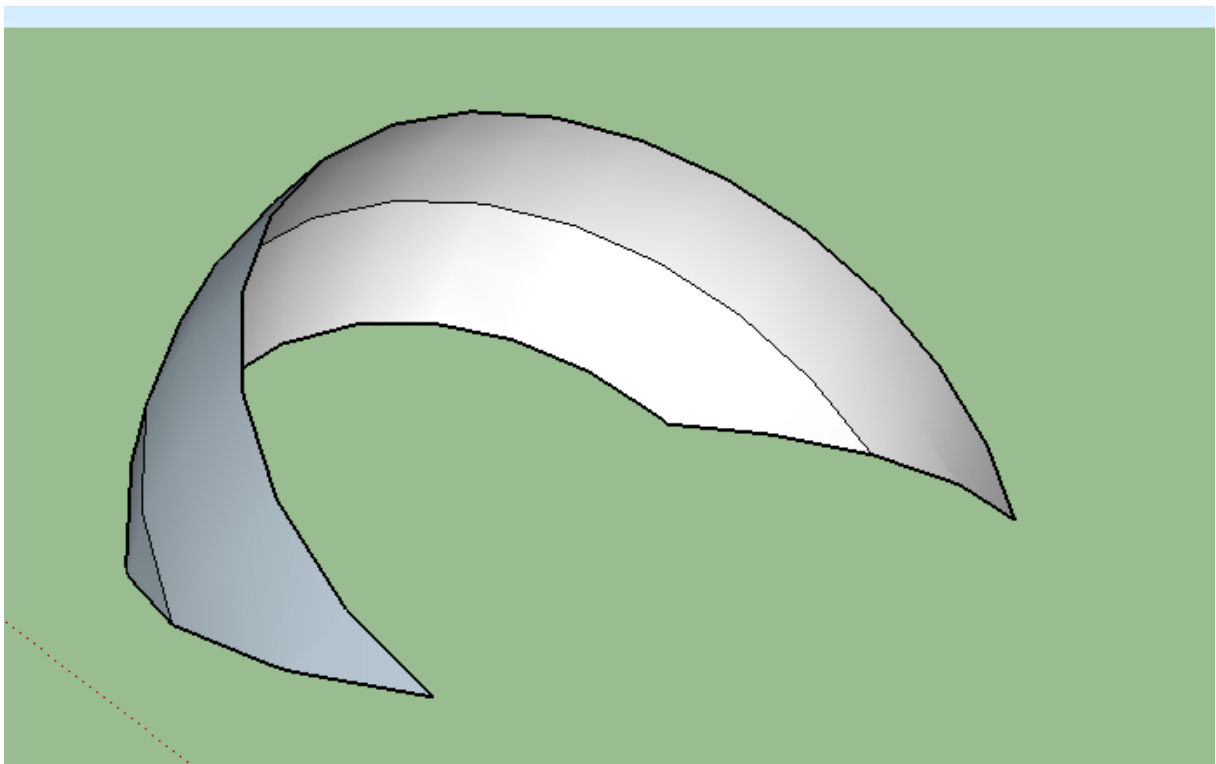
The "Nom de fichier" field contains "milieu urbain" and the "Fichiers de type" dropdown is set to "DAE-Files (*.dae)". The "Enregistrer" button is highlighted. In the background, the application interface shows a sidebar with "ID", "CLASSE", "AIRE", "PERIM", "X", "Y", "LON", "LAT" and a main area with a 3D model of a city. The bottom of the application has a toolbar with buttons: "Importer les valeurs d'un toast", "Supprimer CLASSE", "Modèle 3D à partir de CLASSE", "Ajouter le modèle 2D à un toast", and "Ajouter CLASSE à un toast".

Puis modéliser un modèle 3D en attribuant la couleur des bâtiments en fonction de leur hauteur.

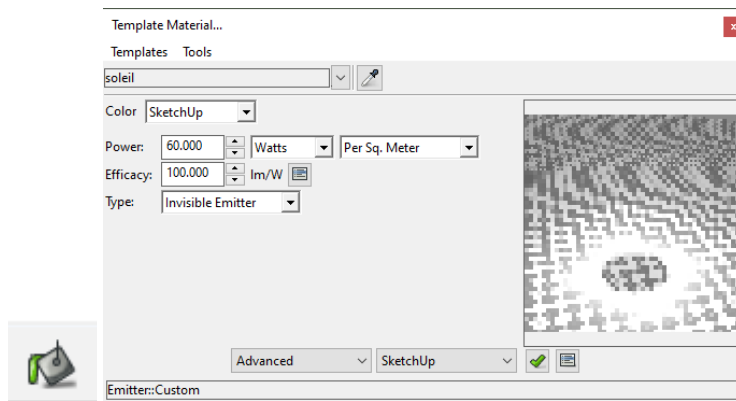


Et importer ce modèle 3D dans SketchUp.

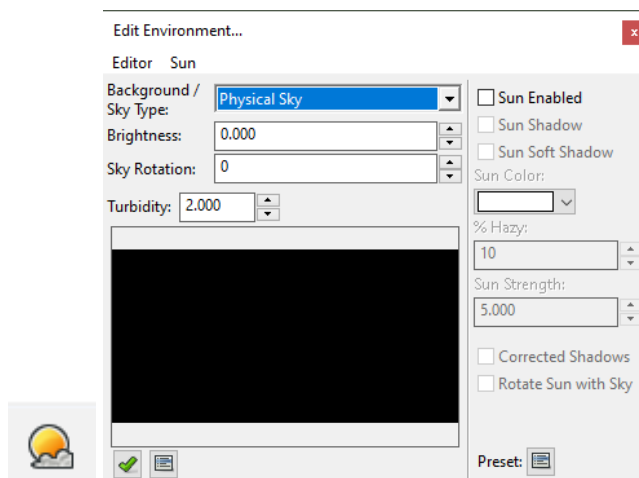
Etape 2 : Diagramme solaire :



Créer un diagramme solaire sur SketchUp avec comme inclinaison 90° moins la latitude du milieu urbain.

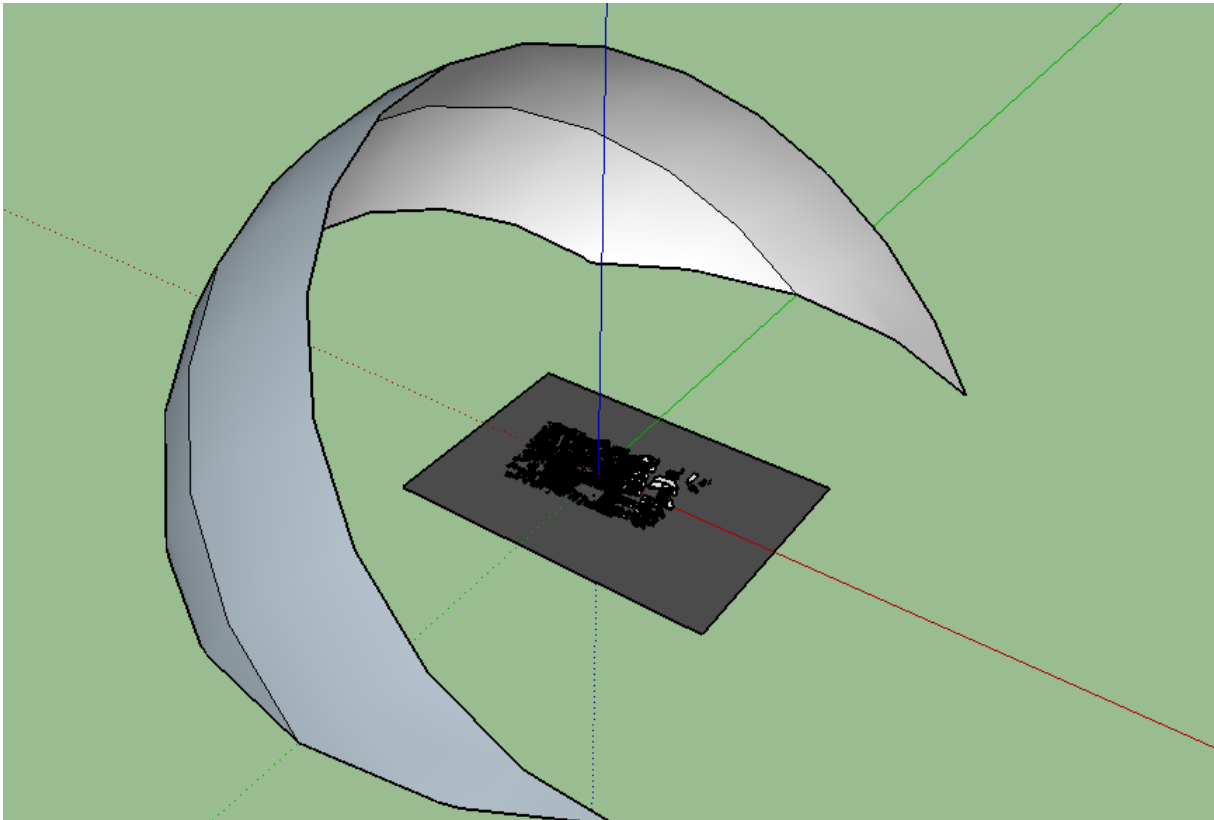


Définir la couleur du diagramme solaire comme un émetteur de lumière selon les paramètres ci-dessus.

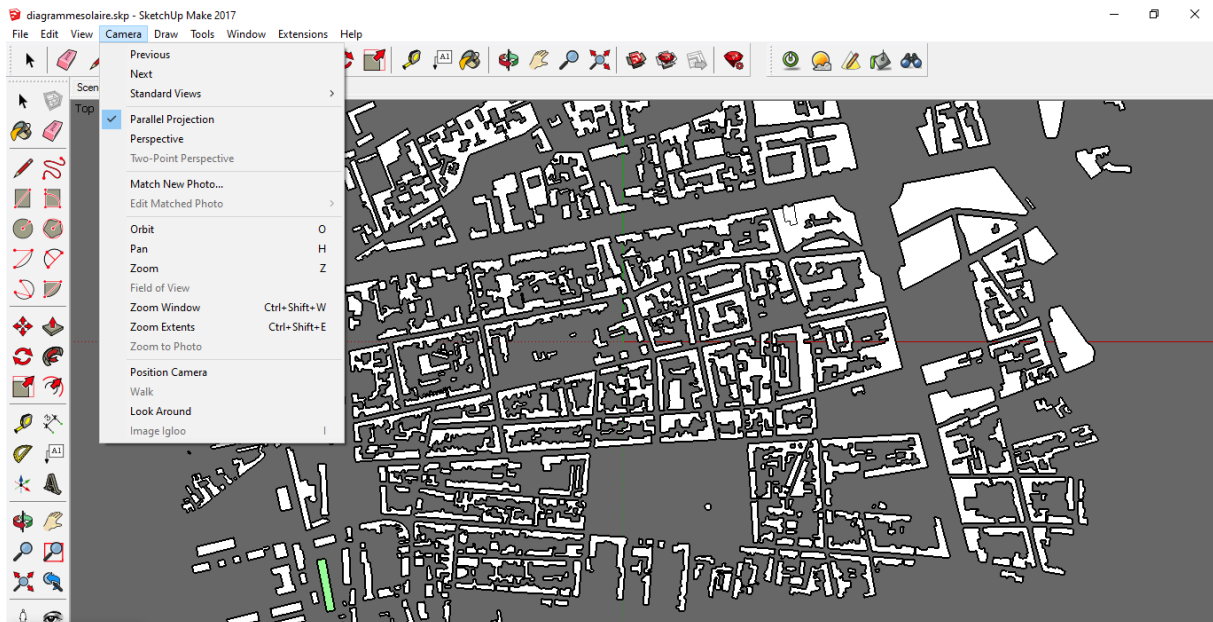


Définir le ciel avec les paramètres ci-dessus.

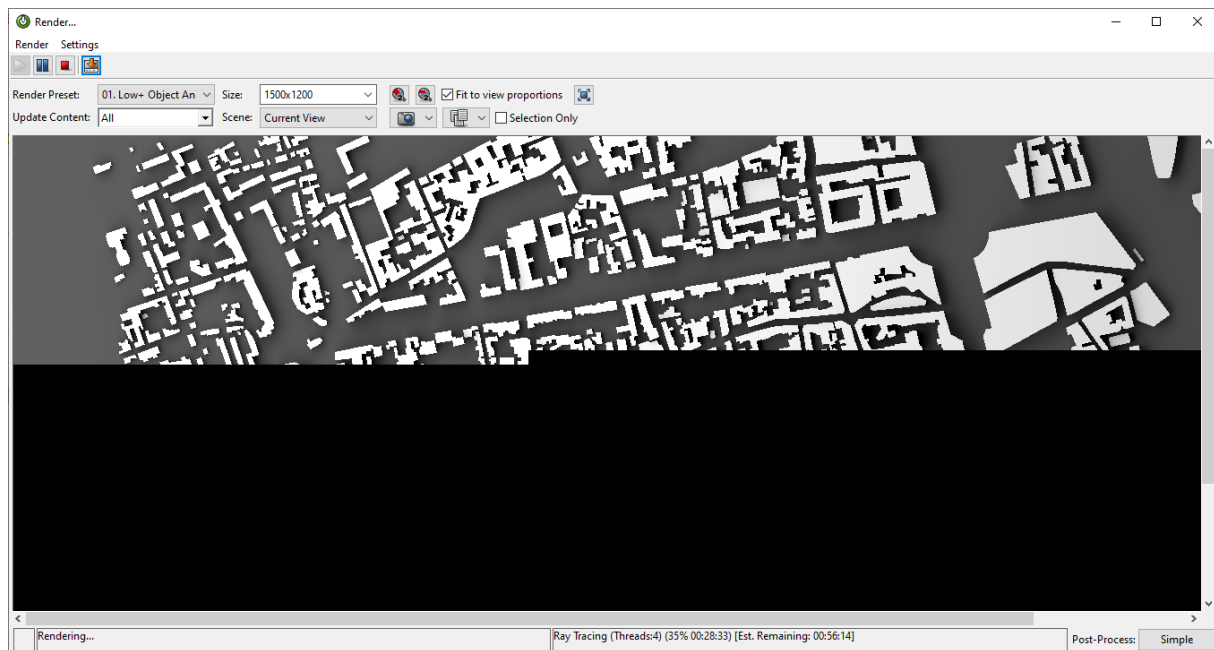
Etape 3 : Simulation, Calcul du rayonnement solaire :



Importer votre milieu urbain au centre du diagramme solaire, et ajouter un sol.



Se placer en vue aérienne, avec la caméra en mode projection parallèle.



Lancer Twilight en mode Low + Object Animation.

Tables de figure :

Figure 1 : Schéma de la réflexion, le faisceau incident va être dévié selon la loi dite de Snell-Descartes.

Figure 2 : Schéma de la réfraction, le faisceau incident va être dévié selon la loi dite de Snell-Descartes.

Figure 3 : Schéma présentant le Ray Tracing et le chemin des rayons simulés.

Figure 4 : Exemple de vue aérienne des bâtiments modéliser (figure de gauche) et de rendu du potentiel solaire (figure de droite) obtenus par Ray Tracing à partir du logiciel SketchUp et de l'extension Twilight.

Figure 5 : Les étapes de la méthode d'apprentissage automatique des k plus proches.

Figure 6 : Exemples d'images du potentiel solaire réalisées grâce à l'algorithme développé dans le cadre du PFE.

Figure 7 : Tableau récapitulatif des 307 images de potentiel solaire réalisé avec l'algorithme du PFE.

Figure 8 : Graphique représentant le temps pour produire une image en fonction de sa taille.

Figure 9 : Exemple de nuages de points représentant la corrélation entre les images obtenues par Ray Tracing et par apprentissage automatique.

Figure 10 : Exemples de répartitions des points de la courbe en selon leurs valeurs pour des images générées avec l'algorithme de Machine Learning.

Directeur de recherche :

Mindjid Maïzia

Paul Vola
PFE/DAE5
UIT/RESEAU
2022-2023

L'intelligence artificielle et le potentiel solaire en milieu urbain : *Comparaison d'un algorithme d'apprentissage automatique avec un algorithme de simulation pour le calcul du potentiel solaire en milieu urbain.*

Résumé :

Les Méthodes de simulation, comme le Ray Tracing, permet un calcul précis du potentiel solaire dans un milieu urbain. Mais cette technique demande un temps de calcul important de l'ordre de plusieurs heures. Afin de déterminer les zones d'installation de panneaux photovoltaïques à fort potentiel il est important de disposer d'une méthode de calcul précise et rapide. Depuis plusieurs années les algorithmes d'intelligence artificielle sont utilisés dans de nombreux domaines dont celui de l'analyse d'images. Lors de ce projet de fin d'étude, j'ai développé un algorithme basé sur le Machine Learning permettant de déterminer le potentiel solaire en zone urbaine, puis je l'ai comparé avec la méthode de Ray Tracing sur les critères de la précision des résultats et celui du temps de calcul. Les résultats de cette comparaison montrent que la précision est comparable mais que les temps de calcul sont environ 100 fois plus rapides en faveur de mon algorithme (à puissance égale).

Mots Clés : Potentiel solaire, Intelligence Artificielle, Ray Tracing, Apprentissage automatique « Machine Learning ».