Projet salamandre

Adrien Chabert, Guy-Raphaël Stauffer et Tommaso Peletta

Application informatique

Professeur accompagnant : Bastien Chopard

Table des matières

Introduction

Ce projet a été réalisé par Adrien Chabert, Tommaso Peletta et Guy-Raphaël Stauffer pour le cours application informatique de deuxième année en bachelor en science informatique de l'Université de Genève. Le professeur accompagnant est M. Bastien Chopard. Ce travail a été fait en collaboration avec la section de Biologie de la faculté des sciences.

L'Hynobius Kimurae, appelée également salamandre d'Hida, est une espèce de salamandre asiatique endémique du Japon qui a des propriétés visuelles très particulières. Les salamandres pondent des œufs dans l'eau pour se reproduire ; ces œufs forment des poches pouvant avoir jusqu'à 70 larves. Les poches d'œufs de cette espèce asiatique ont la particularité d'avoir une couleur bleue quand elles sont au contact de l'eau et une couleur jaune quand elles sont au contact de l'air.

Figure 1. Image de gauche : Poche d'œuf d'Hynobius Kimurae dans l'eau. Image de droite : Poche d'œuf d'Hynobius Kimurae dans l'air

Ce phénomène d'iridescence n'a été que très peu étudié, en particulier en raison de la complexité des structures physique et chimique de ces poches. Il est donc très difficile d'étudier ce phénomène avec des systèmes analytiques devant travailler sur une structure du tissu biologique très complexe et irrégulière. L'approche par procédé analytique nécessite une schématisation de la structure amenant à des résultats trop différents de la réalité du terrain. De plus, la structure qui nous intéresse doit être étudiée avec des outils qui fonctionnent à l'échelle de quelques nanomètres.

Plutôt que d'utiliser une approche analytique, ce projet s'intéresse à la possibilité de simuler le comportement de l'onde lumineuse dans un modèle obtenu à partir de photos prises au microscope du tissu animal, afin d'étudier le spectre lumineux résultant.

Objectifs du projet

Le projet consistait initialement à mettre au point une simulation du comportement d'une onde lumineuse dans le tissu du sac d'œuf d'une salamandre à l'aide de la méthode de Boltzmann sur réseau.

Il s'agissait donc d'implémenter la méthode de Boltzmann sur réseau (LBM), et de la tester avec des exemples dont les résultats sont connus afin de la valider. Dans un deuxième temps, il aurait fallu reproduire exactement la structure du tissu, à l'aide de photos prises au microscope, dans une forme exploitable par la LBM. On aurait ensuite tenté de déterminer s'il est possible d'adapter le maillage et la taille du système afin d'obtenir des résultats cohérents en un temps de calcul raisonnable.

Durant la réalisation du projet, nous avons rencontré des difficultés à valider la LBM dans des exemples simples impliquant la réflexivité de la lumière.

La complexité de la validation de la méthode dépassant le cadre de ce projet, la décision a été prise d'orienter le projet sur le portage des images prises au microscope en un format exploitable par la LBM, laissant à des projet ultérieurs le soin d'adapter la méthode.

L'objectif final de ce projet est donc de fournir un programme permettant de transformer les images du tissu en une matrice d'indices de réfraction en trois dimensions telle qu'utilisée par la LBM.

La méthode de Boltzmann sur réseau.

Communément appelée LBM (lattice-Boltzmann method), la méthode de Boltzmann sur réseau est utilisée pour simuler le comportement dans le temps d'une onde électromagnétique ou d'un fluide newtonien à l'aide d'un maillage. La méthode est appliquée sur un site non-dimensionné. C'est donc à l'utilisateur de définir le pas d'espace δx et le pas de temps δt.

Il est possible de distinguer plusieurs façons d'utiliser ce modèle en fonction du maillage. Une manière pour caractériser ces différentes façons est le schéma DnQm, où "Dn" représente les n dimension du réseau et "Qm" décrit les m directions de propagation de l'onde, l'une de ces directions renvoyant une maille sur elle-même. Ainsi un maillage en deux dimensions où chaque maille est reliée à quatre autres mailles possède en réalité cinq directions et est noté D2Q5.

L'onde est décrite par des quantités fi , avec i = 0 .. m-1, représentant sa distribution de densité.

On distingue deux phases décrites par deux équations qui caractérisent le mouvement de l'onde. Une phase de collision :

$$f_i^{out}(\boldsymbol{r}, t) = f_i^{in}(\boldsymbol{r}, t) + \Omega_i\left(f^{in}(\boldsymbol{r}, t)\right)$$

et une phase de propagation :

$$f_i^{in}(\boldsymbol{r} + \delta_t \boldsymbol{v_i}, t + \delta_t) = f_i^{out}(\boldsymbol{r}, t)$$

qui se répètent à chaque pas de temps.

En particulier pour les cas D2Q5 et D3Q7, choisis pour ce projet, on obtient les équations de collision suivantes :

$$f_i^{out} = \frac{2}{n^2 q}\rho + \frac{1}{v^2}\boldsymbol{v_i} \cdot \boldsymbol{j} - f_i^{in} \quad \text{pour i = 1 .. m-1}$$

$$f_0^{out} = 2\frac{n^2 - 1}{n^2}\rho - f_o^{in}$$

où n est l'indice de réfraction du point traité, $\rho = \sum f_i^{in}$, $v = \frac{\delta_x}{\delta_t}$ est la vitesse du système (à ne pas confondre avec la vitesse de l'onde), vi représente les directions du maillage (avec $|\boldsymbol{v_i}| = v$, pour i = 1 .. m-1) et $\boldsymbol{j} = \sum \boldsymbol{v_i} f_i^{in}$

Un point source d'onde peut être créé à partir de l'équation :

$$f_i^{out}(\boldsymbol{r}, t) = Asin(2\pi v t) \text{ pour i = 1 .. m-1}$$

Où v est la fréquence de l'onde, A est l'amplitude maximale et t est l'itération courante de la méthode multipliée par le pas de temps δt.

Pour simuler des ondes lumineuses, donc unidirectionnelles, on peut implémenter des sources sur une ligne (en deux dimensions) ou un plan (trois dimensions).

Tests de l'implémentation

L'objectif initial du projet ne se limitant pas au portage des données, nous avons testé notre implémentation de la LBM en essayant de reproduire des résultats connus. Ces tests sont effectués en deux dimension (D2Q5), ce qui permet la représentation des résultats en image, et facilite donc leur interprétation.

Pour tester le comportement ondulatoire dans notre système, nous avons simulé trois expériences qui nous donnent effectivement les résultats (observés sur image) attendus :

1. En faisant passer une onde plane (arrivant d'en haut) à travers une couche d'indice de réfraction n = 2, on observe que la longueur d'onde est effectivement divisée par deux.
2. En faisant passer une onde plane (arrivant d'en haut) à travers une couche placée en diagonale, on peut observer qu'une partie de l'onde traverse la couche tandis qu'une autre partie est réfléchie par la couche.
3. En faisant passer une onde émise par une seul source au travers de deux trous, on obtient bien des franges où les ondes sont complétement détruites par les interférences destructives (expérience des fentes de Young).

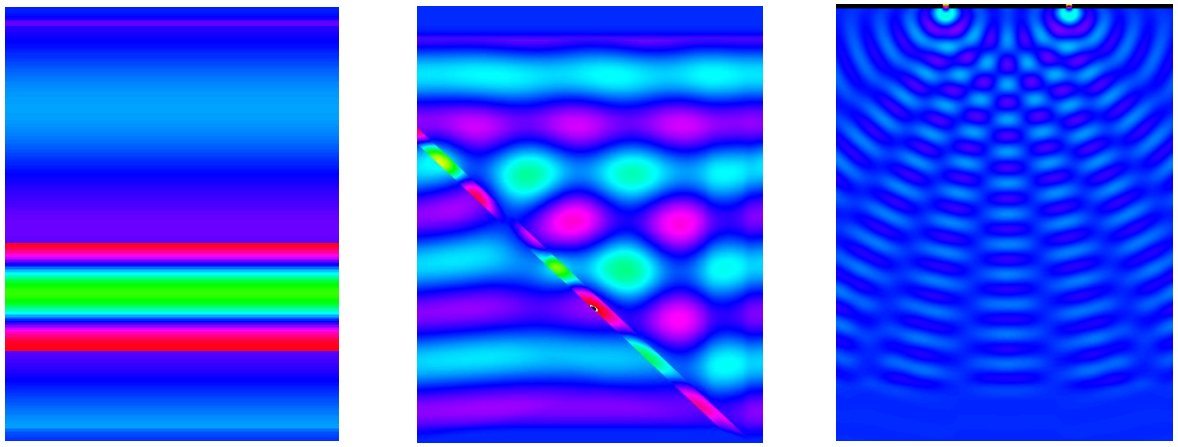

Figure 2: De gauche à droite les expérience 1,2 et 3

Ces expériences attestent le comportement ondulatoire du système.

Ensuite nous avons tenté de reproduire le comportement attendu pour des expériences impliquant la réflexivité (notamment l'expérience de la couche fine). Nous avons alors rencontré des difficultés dépassant la portée de ce projet et avons alors modifié les objectifs du projet, pour nous concentrer sur le portage des données en un format adapté à la LBM.

Portage des données

L'utilisateur fournit au programme des images prises aux microscopes du tissus à analyser. Dans notre cas, il s'agit de photos de couche d'œuf de salamandre d'Hida. Étant donné qu'on traite un problème en trois dimensions, une photo représente seulement un plan du tissu parmi d'autres.

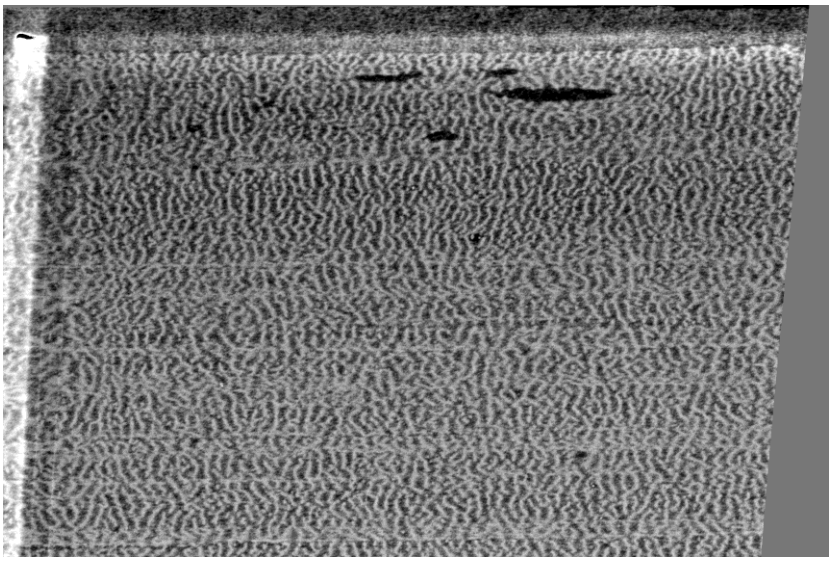

Figure 3 : Hynobius FIB SEM data0000.tif, photo prise au microscope, fourni par Mme Aleksandra Zabuga

Afin d'utiliser la méthode de Boltzmann sur réseau, on doit connaître les indices de réfractions de la matière où l'onde se propage. Ainsi sur notre photo, chaque pixel représente une valeur d'indice de réfraction suivant sa couleur. Dans notre cas, les photos sont en noir et blanc, ainsi chaque pixel peut avoir 256 valeurs possibles, de 0 qui représente le noir à 255 qui représente le blanc.

Pour extraire la valeur de chaque pixel, nous avons utilisé la méthode de partitionnement K-moyennes ou en anglais ''K-mean clustering''. Cette méthode consiste à réunir tous les pixels de toutes les images traitées et à les partager en K groupes différents de façon à minimiser la distance d'un point à la moyenne des points de son groupe.

Dans notre cas, nous voulons partitionner les pixels en 3 groupes (K = 3).

On attribue alors au premier groupe (rassemblant les pixels noirs et gris foncés) l'indice de réfraction de la matière foncée, ici l'eau, donc n = 1.33.

Le deuxième groupe rassemble les pixels blancs et gris clairs, auxquels on attribue l'indice de réfraction de la matière correspondante, dans notre cas, 1.5.

Le troisième groupe est constitué des pixels ayant une valeur de gris intermédiaire, on attribuera alors à chaque pixel une valeur calculée par approximation linéaire entre 1.33 et 1.5.

Calcul du 3-moyennes

La méthode de partitionnement 3-moyennes fonctionne ainsi. Il faut tout d'abord déterminer aléatoirement 3 nombres qui représenteront la position moyenne des 3 groupes (m1, m2, m3). Puis on assigne à chaque pixel le groupe le plus proche par rapport à cette position moyenne.

$$G_i = \{x_j : |x_j - m_i| \leq |x_j - m_{i^*}| \; \forall \; i^* = 1, .., K\}$$

Une fois chaque pixel assigné, on recalcule la positon moyenne des 3 groupes.

$$m_i = \frac{1}{|G_i|} * \sum_{x_j \in G_i} x_j$$

On répète cette opération jusqu'à ce que la position moyenne des groupes ne change pas. Le partitionnement final avec cet algorithme n'est pas toujours optimal et peut avoir un temps de calcul exponentiel suivant le nombre de point. Cependant pour notre problème et notre type de données, il est entièrement suffisant.

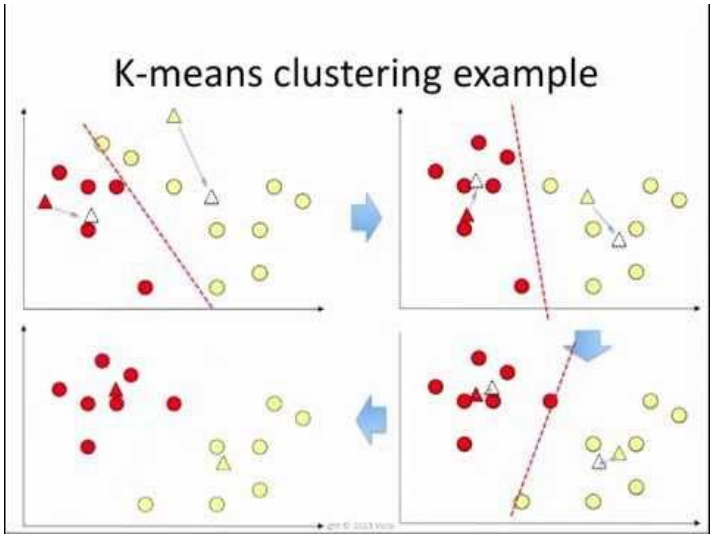

Figure 4 : illustration de la méthode de K-mean Clustering

Une fois calculées ces trois zones de partitionnement, on connaît la valeur des pixels foncés, la valeur des pixels clairs et la valeur des pixels intermédiaires. Ainsi on peut assigner à chaque pixel sa valeur en indice de réfraction.

Prenons comme exemple les photos d'œufs de salamandre qui nous ont été fourni. On applique l'algorithme sur 5 images. Le graphe ci-dessous représente l'histogramme de ces 5 photos.

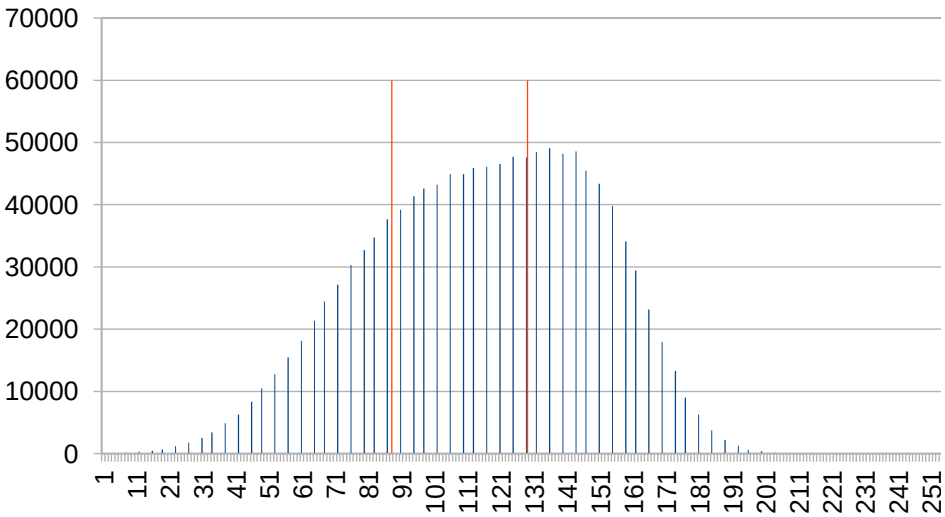

Figure 5 : Histogramme de 5 photos, en orange figure la limite des partitions suivant l'algorithme de partitionnement 3-moyennes.

On obtient des positions moyennes en 68.04, 109.39, 149.55. En orange est représenté la limite des partitions. La partie de gauche représente les pixels foncés qui auront une valeur de 1.33. La partie de droite représente les pixels clairs qui auront une valeur d'indice de réfraction de 1.5. On applique à la partie du milieu, une approximation linéaire allant de 1.33 à 1.5 suivant la valeur du pixel d'origine.

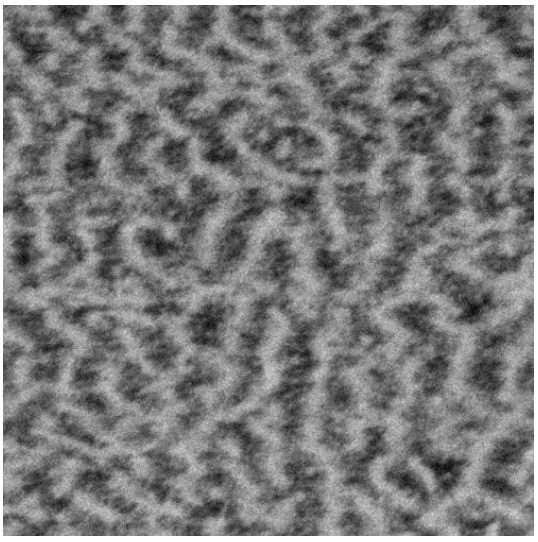

Figure 6 : Image d'origine redimensionner 500x500

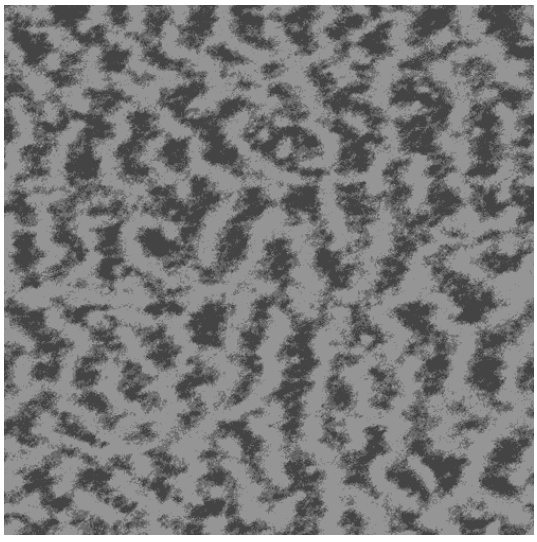

Figure 7 : Image représentant les valeurs d'indice de réfraction après l'algorithme 3-moyennes

Interpolation linéaire entre les couches

La distance entre les différentes images prises, c'est-à-dire la distance entre chaque couche prise en photo (espacement des plans sur l'axe des Z), ne correspond pas toujours à la précision des pixels de l'image, qui correspond à l'espacement des points sur l'axe X et Y. Par exemple dans notre cas, la précision sur les axes X et Y est de 3,9 nanomètres alors que l'espace entre chaque photo est de 10 nanomètres. Ceci pose problème lorsqu'on applique la LBM : l'onde se propagera plus rapidement sur l'axe des Z que sur les axes X et Y.

Il est nécessaire d'ajuster les précisions. Il faut que la précision sur les trois axes soit la même. Pour ajuster la précision sur l'axe des Z, on a utilisé une approximation linéaire. Dans notre cas, l'approximation linéaire a dû ajouter des points sur l'axe des Z afin d'avoir une précision de 3,9 et non 10 nanomètres (voir figure 8). C'est pourquoi, à la fin de l'exécution du programme, il y a plus de couches d'indice de réfraction créés que de nombre d'image traitées.

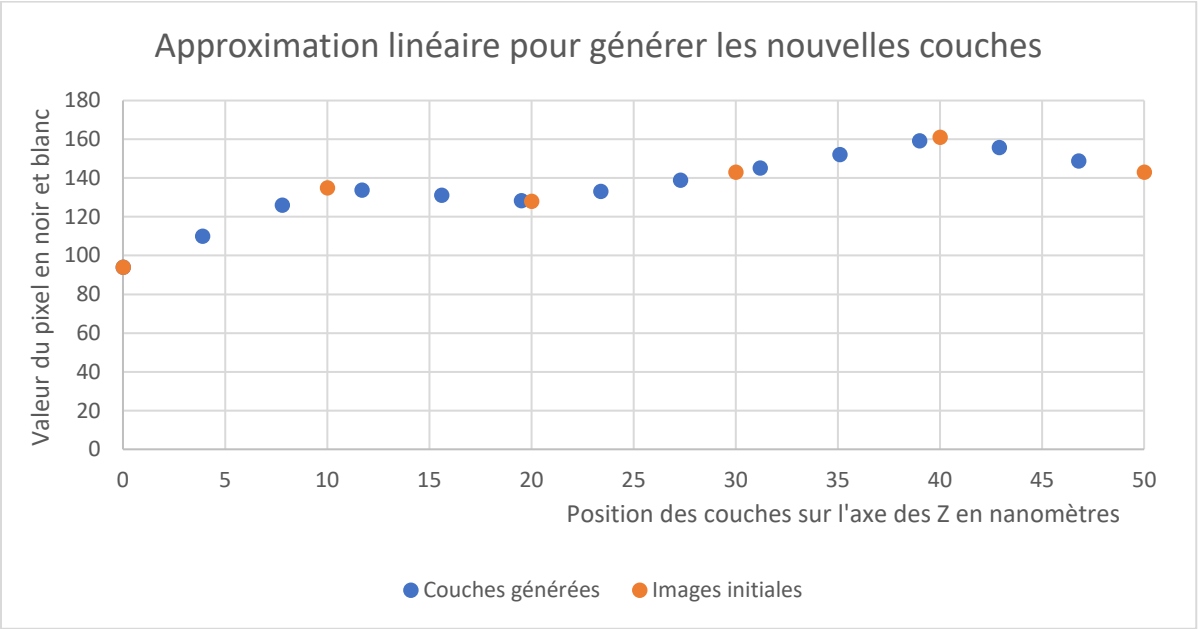

Figure 8 : Illustration de l'approximation linéaire sur un point. Espacement entre les couches : 10nm. Précision voulue : 3,9 nm. 6 images étudiées. 13 couches générées.

Utilisation du programme

Le portage des données dans une matrice exploitable par la LBM se fait à l'aide de deux programmes.

Un programme python : Refraction_Indexes_Converter.py, qui lit les images fournies, calcul le coefficient de réfraction de chaque maille à l'aide de la méthode des 3-moyennes et de l'interpolation linéaire, et stocke les valeurs dans un fichier logfile.txt

Un programme C++ : 3d_wave.cpp, qui récupère les données du fichier lofile.txt et les utilise pour remplir une matrice, qui pourra alors être utilisée par le programme avec la méthode de Boltzmann sur réseaux.

Le programme LogFileReader.py permet d'afficher une couche du logfile.txt.

Refraction_Indexes_Converter.py :

Refraction_Indexes_Converter.py est un fichier qui extrait les indices de réfraction de plusieurs couches du tissu des poches d'oeufs de salamandre Hida à partir d'une image.

Le programme écrit les données collectées dans un fichier logfile.txt.

Une fois le code exécuté, l'utilisateur doit rentrer les paramètres demandés par la GUI, en particulier :

- StartX : le point de départ sur l'axe x pour définir la zone de l'image à considérer
- StartY : le point de départ sur l'axe y pour définir la zone de l'image à considérer
- EndX : le point sur l'axe x qui définit la fin de la zone de l'image à considérer
- EndY : le point sur l'axe y qui définit la fin de la zone de l'image à considérer
- Distance between two pixels of the image : définit la distance en nanomètres entre deux pixels
- Distance between two layers : définit la distance en nanomètres entre deux couches
- White refraction index : indice de réfraction de la partie plus claire de l'image
- Black refraction index : indice de réfraction de la partie plus sombre de l'image
- Number of images : nombre d'images dont on doit effectuer la conversion

Exécution du programme :

- Pour Linux

  Le code peut être exécuté directement à partir du fichier exécutable Refraction_Index_Converter. Il peut aussi être exécuté depuis la ligne de commande : python3 Refraction_Index_Converter.py

- Pour Windows

  Le code peut être exécuté à partir de python IDLE (https://www.python.org/downloads)

Prérequis pour exécuter le programme (pas requis si lancé depuis l'exécutable) :

- librarie PIL
- librarie PyQt4 (sudo apt-get install python3-PyQt4)

Ecriture du logfile :

Les premières lignes du logfile contiennent les dimensions de la matrice (sizeX, sizeY, sizeZ), puis les valeurs utilisées pour le calcul du coefficient de réfraction :

- bornInf et bornSup sont les indices de réfractions des pixels foncés et clairs respectivement.
- pMin et pMax sont les valeurs frontières calculées par la méthode 3-moyenne, qui séparent les trois groupes de pixels.

Viennent ensuite les indices de réfractions regroupés par couches. Chaque couche contient une ligne par valeur de y (sizeY lignes), et chaque ligne contient un indice de réfraction par valeur de x (sizeX indices par couche). Les indices sont séparés par des espaces " ".

Attention !

L'exécution du code se termine par l'affichage d'une fenêtre pop-up.

Si le programme est lancé depuis la console, la progression est affichée dans la console.

Si le programme est lancé en utilisant l'exécutable, la progression n'est pas affichée, il faut donc attendre l'apparition de la fenêtre pop-up pour savoir que le programme est terminé, ce qui peut prendre longtemps.

LogfileReader.py :

LogfileReader.py est un fichier qui permet d'afficher une image qui correspond à une couche de la matrice des indices de réfraction contenue dans le logfile.txt.

Le programme demande à l'utilisateur de choisir la couche à traiter. Il affiche cette dernière en transformant les indices de réfraction en valeur de pixel noir et blanc. Veuillez prendre en compte que la valeur des pixels clairs prend pour valeur la moins élevée possible (la valeur la plus sombre pour leur partie). Alors que les pixels foncés prennent la valeur la plus élevée (la valeur la plus clair pour leur partie). Ceci explique le faible contraste de l'image crée. Veuillez consultez la figure n°5 pour plus de compréhension.

Exécution du programme :

- Pour Linux
  Le code peut être exécuté depuis la ligne de commande : python3 LogfileReader.py
- Pour Windows
  Le code peut être exécuté à partir de python IDLE (https://www.python.org/downloads)

Prérequis pour exécuter le programme :

- Librairie PIL
- Librairie PyQt4 (sudo apt-get install python3-PyQt4)

3d_wave.cpp :

3d_wave.cpp est un fichier qui recueille les données extraites des images dans le fichier logfile.txt et les stocke dans une matrice tabl_n. Les dimensions de la matrice sont extraites directement depuis logfile.txt.

Le fichier contient aussi la méthode de Boltzmann sur réseaux en trois dimensions que nous avons implémentée.

Les conditions aux bords sont gérées dans la procédure vector_cpy et à l'aide de la matrice d'atténuation beta. Par défaut, ces conditions sont implémentées de façon à être périodiques dans les trois directions, donc sans atténuation.

Les points de la matrice tabl_n possédant un indice de réfraction strictement compris entre 0 et 1 sont considérés comme des sources, tandis que les points dont la valeur est inférieure ou égale à 0 sont considérés comme des miroirs.

La méthode de Boltzmann s'applique sur la matrice tabl_n qui contient les valeurs des indices de réfraction extraites de logfile.txt. Les données extraites n'ayant pas de source, la méthode ne fait rien en l'état.

Il est aussi possible d'utiliser ce fichier sans les données du logfile. Il faut alors retirer la procédure Hynobius() de la fonction main, les dimensions de la matrices sont déterminées dans la procédure allocate().

Pour compiler le programme, il faut utiliser la commande : g++ 3d_wave.cpp -o 3d_wave

Il faut alors s'assurer que le fichier logfile.text se trouve dans le même dossier que l'exécutable 3d_wave (sauf si on veut utiliser le programme sans les données d'un logfile).

On peut alors lancer le programme avec la commande : ./3d_wave

Conclusion

Notre objectif initial était d'étudier la réflexivité de la lumière sur les structures des œufs de salamandre à l'aide d'une simulation informatique utilisant la LBM.

Il a fallu de multiples tentatives infructueuses pour arriver à la conclusion que cette étude ne pouvait être menée à son terme à cause de problèmes physiques dépassant le cadre de ce cours.

Ceci nous a malheureusement privés de l'intérêt informatique d'un projet de simulation comme la gestion de la taille du système, de la précision du maillage et du temps de calcul et nous le regrettons.

Nous espérons malgré tout que le programme proposé pour intégrer les données dans une matrice adaptée à la méthode de Boltzmann sur réseaux sera utile, une fois la méthode validée, et qu'il aidera, à terme, à mieux étudier les phénomènes d'iridescence.