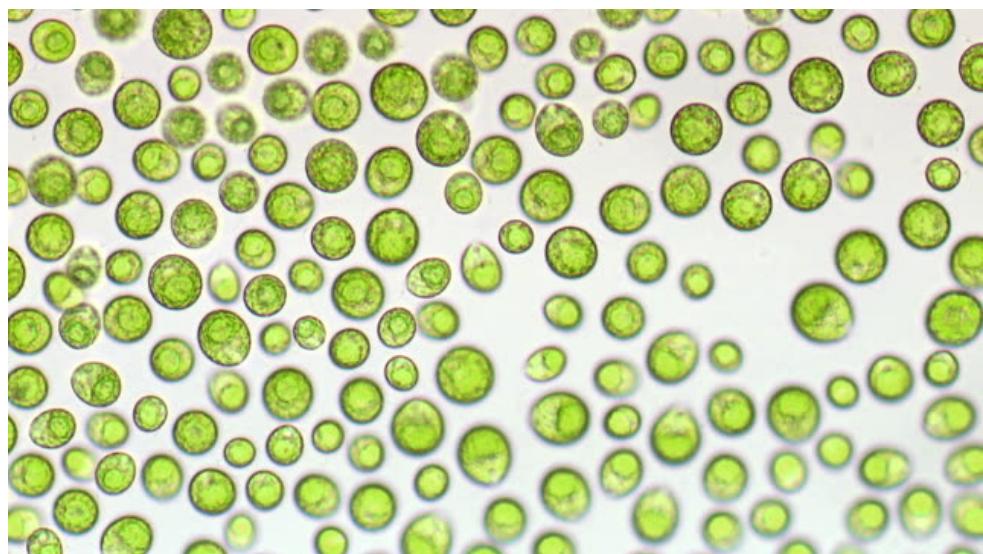


Nicolas SALVAN - Thomas CHEN
Romain CAPRON - Merlin FOUCHÉ
Antoine POULAIN - Clément ZHUANG

MAIN 3
2022-2023
Projet pluri-disciplinaire

RAPPORT DE PROJET

Modélisation et analyse du phénomène d'agrégation chez les algues unicellulaires
Chlamydomonas reinhardtii



POLYTECH[®]
SORBONNE



SORBONNE
UNIVERSITÉ

Enseignants encadrants : Antoine DANON
Thibault HILAIRE

Tout d'abord, nous tenons à remercier Monsieur Thibault HILAIRE pour nous avoir proposé de réaliser un tel projet.

Nous souhaitons ensuite remercier Monsieur Antoine Danon de nous avoir soumis ce sujet singulier, mais pour autant très intéressant. Nous souhaitons également souligner son accueil, son soutien, et son accompagnement au cours de sa réalisation.

Merci à nos enseignants.es de MAIN3 et des années passées de nous avoir permis d'acquérir les connaissances nécessaires à la réalisation de ce projet, et l'envie de découvrir celles encore inconnues.

Ce rapport est le fruit de notre travail dans le cadre du projet pluridisciplinaire proposé par notre enseignant M. Thibault HILAIRE au cours de notre formation en Mathématiques Appliquées et Informatique Numérique (MAIN 3) à Polytech Sorbonne.

Table des matières

Introduction	3
1. Modélisation	4
Création de la zone de simulation	4
Création de la population d'algues	5
Principe de la modélisation	6
Déplacement des algues	6
Division des algues	6
Suppression des cellules	7
Modélisation des agrégats	8
Création d'un GUI pour paramétriser la simulation	10
2. Analyse d'image	12
Premier pas, première méthode	12
Problèmes rencontrés	16
Conclusion	17
Annexes	18
1. Code du projet	18
2. Sources	18
3. Analyse d'image	18
4. Interface graphique	21

Introduction

Dans l'immensité du règne végétal planétaire, on peut compter les *Chlamydomonas reinhardtii*. Ces êtres vivants dont la taille se compte en une dizaine de micromètres sont des algues vertes unicellulaires qui nagent grâce à deux flagelles extra-cellulaires.

Cette algue est aujourd'hui étudiée dans le cadre de recherches sur la photosynthèse, de la synthèse de protéines et de la mobilité induite par les flagelles.

Dans d'autres cadres, ces algues sont aussi étudiées pour leurs possibles bénéfices industriels. En effet, ces algues pourraient jouer un rôle important dans la production de biocarburants à un moindre coût par rapport aux coûts des techniques considérées comme traditionnelles.

Les recherches précédentes avaient remarqué que si ces algues sont exposées à un stress abiotique au cours de leur division, elles sont amenées à former des palmelloïdes.

Aujourd'hui, une autre réaction de cette algue à un stress fait l'objet de recherches particulièrement actives. En effet, les *Chlamydomonas reinhardtii* peuvent, si soumises à un stress abiotique approprié, former des agrégats de cellules. Les études faites à ce sujet s'intéressent notamment au lien entre cette réaction et la formation d'êtres multicellulaires à partir d'algues unicellulaires.

L'objectif de ce projet, proposé par M. Antoine Danon, est de modéliser l'agrégation des algues vertes (*Chlamydomonas reinhardtii*) et de s'intéresser à l'importance du paramètre d'adhésion des cellules dans ce modèle, ainsi que de vérifier la véracité de notre modèle.

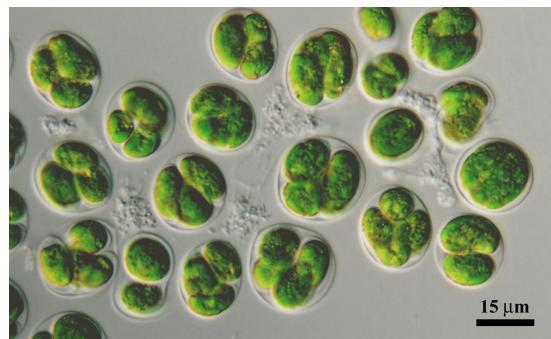


Figure 1 : microphotographie de palmelloïdes des algues Chlamydomonas reinhardtii qui restent unies après division

Pour réaliser ce projet, nous nous sommes répartis les tâches entre la création et l'implémentation du modèle et l'analyse des images et vidéos des expériences conduites par M Antoine Danon. Étant donné la récence de ces recherches sur ce phénomène d'agrégation, la quasi-totalité de nos données nous ont été transmises par notre encadrant.

La question du langage de programmation a imposé le langage Python comme étant le plus optimal, notamment pour les nombreuses bibliothèques écrites pour ce langage. Pour

mettre en commun notre code, nous avons créé un dépôt GitHub dont le lien se trouve en annexe ([Annexe 1](#)).

Nous aborderons dans un premier temps notre démarche pour la concrétisation de ce modèle ainsi que les différents obstacles que nous avons rencontrés. Dans un second temps, nous nous pencherons sur la vérification de ce modèle réalisée par une analyse d'image.

1. Modélisation

L'agrégation des algues *Chlamydomonas reinhardtii* est un phénomène qui dépend principalement du type et de l'intensité du stress auquel on les soumet. Les théories sur le déroulement de l'agrégation de ces cellules sont variées, mais une hypothèse que nous avons essayé de modéliser est que les algues peuvent se coller au contact d'autres algues lorsqu'elles sont exposées à du stress abiotique.

Afin de simuler un grand nombre d'objets (les algues), nous avons décidé d'utiliser les bibliothèques *Matplotlib* pour l'affichage de la simulation, et *Numpy* pour la réalisation des calculs. Nous avons également utilisé de la programmation orientée objet pour modéliser des objets comme la population d'algues, et la zone de simulation. Afin d'optimiser le calcul de la simulation et d'améliorer la compréhension de notre code, nous avons procédé à une programmation modulaire.

Ainsi, nous avons commencé par créer les objets et implémenter les fonctions essentielles. Nous nous sommes ensuite intéressés à la création d'agrégats, et enfin à l'interface graphique permettant de faciliter la saisie des paramètres.

Création de la zone de simulation

```
Class Box(x_max, y_max)
```

Pour créer la zone de simulation, nous avons créé une classe *Box* représentant l'abscisse et l'ordonnée maximale des algues. Etant donné que les coordonnées des algues sont positives, nous n'avons pas besoin de stocker plus d'information dans cette classe. Pour le créer, il suffit de rentrer en argument la taille de la zone de simulation en unités (1 diamètre de cellule représente une unité dans notre simulation).

Attributs de la classe <i>Box</i> (x_max, y_max)		
Nom	Type	Description

x_max	float	Représente l'abscisse maximale.
y_max	float	Représente l'ordonnée maximale.

Tableau 1 : Attribut de la classe Box

Cette classe permet de définir les dimensions de la zone dans laquelle aura lieu la modélisation

Création de la population d'algues

Class Population (nombre_algues, Box)

Après plusieurs essais, nous avons décidé de manipuler l'entièvre population d'algues plutôt que plusieurs objets algues afin de pouvoir définir des fonctions plus concises. Nous avons ainsi choisi de créer une classe Population. L'état de la population d'algues est accessible via ses attributs, des tableaux numpy. Ainsi chaque algue est identifiée par un indice i, à partir duquel nous pouvons obtenir les informations souhaitées.

Par exemple, si nous voulons vérifier l'âge de la 4e cellule, nous obtenons la donnée via l'expression *population.age[4]*.

Attributs de la classe <i>Population (nombre_algues, Box)</i>		
Nom	Type	Description
nombre_algues	int	Nombre d'algues vivantes de la population.
x	array de float	Abscisse x des algues. Compris entre 0 et la valeur x_max de la Box.
y	array de float	Ordonnée y des algues. Compris entre 0 et la valeur y_max de la Box.
agregat	array de booléens	Etat d'agrégation des algues. S'il vaut True alors l'algue est agrégée, si il vaut False alors elle ne l'est pas.
age	array d'entiers	Etat de reproduction des algues. Pour chaque cellule, c'est un nombre entre 0 et le nombre de frames qu'il faut pour que l'algue ait un âge suffisant pour pouvoir se diviser.
taille	array d'entiers	Taille de chaque algue. Représente la taille de l'algue, parmi 4 tailles : TAILLE_1 pour une seule algue, TAILLE_2, TAILLE_4 et TAILLE_8 respectivement pour des palmelloïdes de 2, 4, 8 cellules.

Tableau 2 : Attributs de la classe Population

Différents attributs permettant de définir la population d'algues présente dans la simulation

Principe de la modélisation

L'évolution de la population se fait en deux étapes. Dans un premier temps, on procède à l'initialisation de la population. Puis, pour un certain nombre d'itération choisi par l'utilisateur, on met à jour la population d'algues. On prend alors en compte les déplacements, la division des algues et les autres paramètres.

Pour l'initialisation, nous utilisons simplement les classes Box et Population. Pour l'actualisation de la population, on utilise les modules *SIM_deplacement.py*, *SIM_reproduction.py* et *SIM_mort.py*. Par la suite, nous avons ajouté l'agrégation des cellules avec le module *SIM_agregat.py*, et les paramètres de la simulation dans le module *_settings.py*.

Quant au rendu final de la simulation, nous avons décidé de créer une vidéo ou un gif en utilisant la librairie *matplotlib.animation*, qui nous permet de générer une animation et de la sauvegarder au format .mp4 ou .gif. Nous avons utilisé la bibliothèque ffmpeg pour enregistrer la vidéo au format mp4

Déplacement des algues

module *SIM_deplacement.py*

Dans notre simulation, le déplacement des cellules se fait selon une marche aléatoire grâce à l'outil *randint* de la bibliothèque *random*. Ainsi, nous générons une valeur aléatoire comprise dans l'intervalle $[-DEP_MAX, DEP_MAX]$ pour x et y, et l'ajoutons à la position x et y de la cellule. Ici, *DEP_MAX* est le déplacement maximal que la cellule peut effectuer entre deux itérations de temps. Elle peut alors se déplacer dans le carré dont elle occupe le centre, de coté $2*DEP_MAX + 1$.

Dans ce modèle, nous devons prendre en compte les collisions que ces déplacements peuvent occasionner. Pour ce faire, nous avons créé une fonction *check_deplacement*, qui nous permet de savoir si le déplacement est possible ou non, en vérifiant si une cellule est déjà présente à l'endroit où on veut déplacer l'algue, et en vérifiant que l'algue ne sort pas de la zone de simulation. Nous avons ensuite défini une fonction *deplacement_cellule* qui génère le déplacement en mettant à jour le tableau des positions pour les cellules non agrégées.

Ce modèle de déplacement permet une plus grande vitesse de calcul mais il est moins précis : si l'algue se déplace de $(+DEP_MAX, +DEP_MAX)$ par exemple, on voit bien qu'elle parcours une plus grande distance que *DEP_MAX*.

Division des algues

module *SIM_reproduction.py*

Afin de modéliser la division des algues vertes, il nous a fallu dans un premier temps augmenter la taille des cellules afin d'illustrer le phénomène de multiplication. Nous avons alors créé une fonction *augmenter_taille*, et introduit un nouveau paramètre appelé *TEMPS_REPROD*, qui correspond au nombre de frames entre chaque augmentation de taille correspondant à une division cellulaire.

Par la suite, nous avons défini deux autres fonctions. La fonction `is_room_for_reprod` permet de savoir si une algue dispose de suffisamment d'espace pour se diviser. Elle renvoie True si il y a assez de place autour de la cellule pour se diviser, renvoie False sinon. Nous avons utilisé la fonction `check_deplacement` pour pouvoir analyser le voisinage de la cellule.

La seconde fonction, `reproduction`, permet d'appliquer la division sur les cellules pouvant se diviser. Elle crée alors 7 cellules filles et modifie les informations de la cellule mère dont sa taille.

Suppression des cellules

module `SIM_suppression.py`

La suppression des cellules dans la simulation représente leur mort. Afin d'être proche de la réalité et sur recommandations de notre encadrant, nous avons décidé de ne tuer que les cellules non agrégées et de le faire en fonction d'une probabilité donnée par le stress (`settings.STRESS`).

Le stress est modélisé uniforme sur l'ensemble du plan de simulation c'est-à-dire que, quelle que soit la position de la cellule dans la Box, elle sera soumise au même stress. Celui-ci est modélisé par une probabilité de vie ou de mort par le biais d'un schéma de Bernoulli dont la probabilité de mort est, dans ce cas, le stress appliqué. Ce schéma est appliqué à chaque cellule non-agrégée, à chaque frame.

Un des principaux enjeux de cette représentation est de réussir à équilibrer la fréquence de division des algues et leur taux de mortalité. Dans le cas où cet équilibre n'est pas atteint, il se peut que toute la population meurt avant d'avoir eu la possibilité de se reproduire, ou bien qu'il n'y ait pas assez de morts et le nombre de cellules explose.

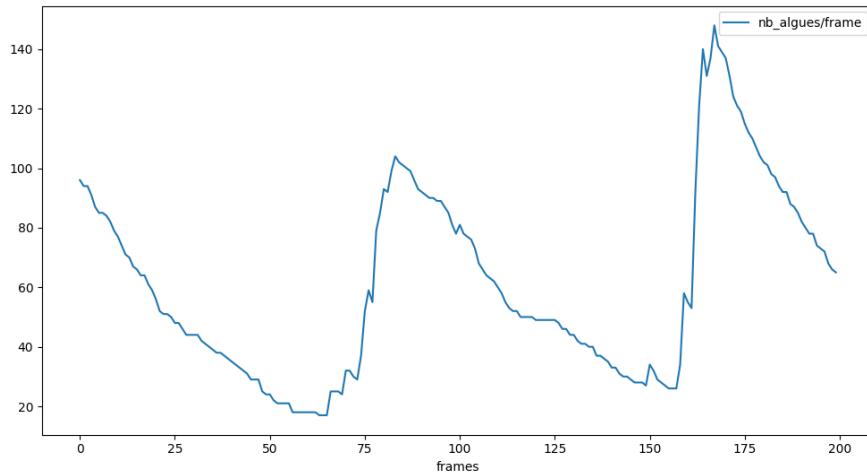


Figure 2 : Evolution du nombre d'algues en fonction du nombre de frames lorsque seul les suppressions et les reproductions sont appliquées au modèle

Modélisation des agrégats

Avant de parvenir à réaliser un modèle cohérent, nous avons rencontré une variété de difficultés que nous avons dû appréhender, et nous avons ainsi produit différentes manières de modéliser l'agrégation .

Dans une première tentative, afin de simuler la formation d'agrégat, nous nous sommes basés sur le principe stress non uniforme, avec les zones d'agrégation confondues avec les zones de stress.

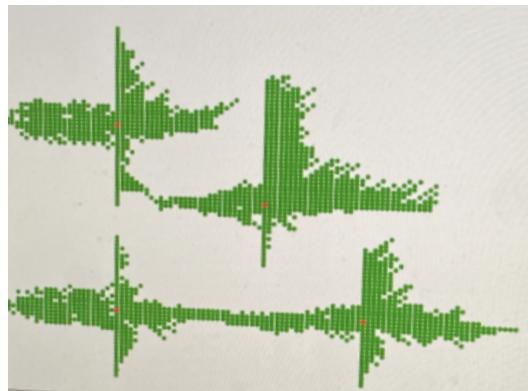


Figure 3 : Première simulation des agrégats

Photo de la première modélisation des algues (points verts) agrégés autour de points de stress (points rouges) choisis de manière non uniforme et aléatoire

Sur la Figure 3, les points rouges correspondent ainsi aux zones de stress élevé. Nous avons choisi ce modèle suite à l'évocation de notre tuteur du principe de matrice collante, une matière extracellulaire produite par les algues agrégées permettant l'adhésion et l'apport de nutriments aux autres algues.

Après une longue réflexion et une étude approfondie de plusieurs articles évoquant l'agglomération de cellules *Chlamydomonas reinhardtii*, nous nous sommes rendu compte que notre système n'était pas cohérent avec les observations.

En effet, les expériences mentionnent un stress appliqué de manière uniforme sur l'ensemble de l'environnement. Ainsi les agrégats se génèrent de manière aléatoire, sont en mouvement et sont de tailles variables en fonction du stress.

Nous avons alors décidé de voir le problème sous un autre angle et de produire une nouvelle structure pour notre simulation.

De ce fait, nous nous sommes basés sur le fait que dans un environnement de stress uniforme, l'ensemble des cellules vont alors produire une matière collante autour d'elles, afin de se regrouper et ainsi, créer des agrégats. De plus, plus un agrégat est grand, plus la quantité de matière produite est grande.

Pour représenter ce phénomène, nous avons alors créé deux nouvelles probabilités. La première, PROBA_AGREGAT, représente le pourcentage de chance qu'une cellule s'agrège à un agrégat déjà existant. Les cellules n'étant initialement pas agrégées, nous avons alors posé une seconde probabilité PROBA_AGREGAT_NORMAL qui désigne le pourcentage de chance que deux cellules libres s'agrègent entre elles.

Nous avons ensuite défini une fonction `update_agregat`, qui applique les deux probabilités précédentes selon une loi uniforme aux cellules de notre environnement, et génère des agrégats dans un tableau nommé `agregat`.

Enfin, nous avons créé une fonction `deplace_agregat`, qui permet de générer un déplacement pour les agrégats, en partant du même principe que le déplacement d'une cellule, la marche aléatoire, mais cette fois en parcourant la liste `agregat`.

Afin d'évaluer notre simulation, nous l'avons comparé aux observations en laboratoire.

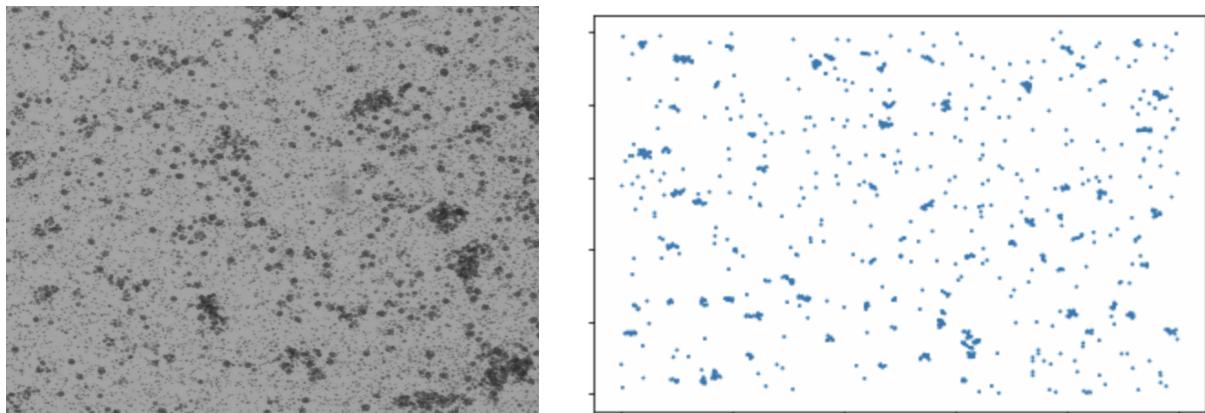


Figure 4 : Cellule de *Chlamydomonas reinhardtii* dans un environnement de stress en simulation et en expérience réelle

Comparaison entre les résultats réels (à gauche) et modélisés (à droite). Les points noirs de la figure de gauche sont les algues vertes vivantes, les points bleus de la figure de droite sont les algues vertes modélisées.

Dans la Figure 4 ci-dessus, nous pouvons remarquer des agrégats de points bleus formés après avoir appliqué du stress aux algues. Ces structures correspondent à des cellules qui se sont regroupées pour former des agrégats. Ainsi, en comparant nos résultats recueillis via la simulation aux expériences, nous pouvons noter une cohérence du modèle.

Bien que ce modèle soit cohérent avec la réalité, il reste des pistes d'amélioration à notre simulation.

La première d'entre elles est l'optimisation du temps de calcul. En effet, même si notre simulation est réussie et nous génère un gif ou une vidéo selon ce qui lui est demandé, le temps de calculs nécessaires à l'implémentation du modèle reste plus que conséquent. Notamment, pour générer une simulation de 20 000 algues, notre modèle a produit une vidéo de 30 secondes en une quinzaine de minutes de calculs.

Un autre point non abordé dans ce modèle est l'implémentation de collision. Afin d'éviter de générer des collisions entre les différentes cellules de notre environnement, nous avons créé la fonction appelée `check_deplacement`. Cependant, cette fonction vérifie simplement si une cellule est présente ou non à la position où veut se déplacer la cellule de départ. Les collisions ne sont donc pas correctement implémentées. En effet, si une cellule est présente sur le chemin de l'algue, le déplacement sera quand même effectué sans prendre en compte la rencontre éventuelle avec cette cellule .

Finalement, la simulation ne prend pas en compte la rotation des agrégats. Dans notre simulation, nous avons réussi à créer une fonction qui simule le déplacement des agrégats mais pas leur rotation. En effet, l'implémentation de cette caractéristique demanderait à l'ordinateur une puissance de calculs d'autant plus importante, et ainsi augmenterait le temps de rendu d'un résultat.

Paramètres de la simulation

modules GUI_<nom_du_module>.py et _Main_Window.py

Une telle simulation nécessite un grand nombre de paramètres différents, allant de la densité initiale d'algues à la taille de la boîte utilisée, en passant par le niveau de stress appliqué. Ne pouvant nous résoudre à fixer arbitrairement ces réglages, nous avons décidé de proposer une interface graphique laissant le choix des valeurs. Le terme GUI se réfère à l'expression anglaise Graphical User Interface.

Tableau des paramètres entrés par l'utilisateur		
Paramètres sur les algues	[min - max] unité	Signification
Nombre d'algues initiales	[10 - 2000] unit	Nombre d'algues initiales
Vitesse des algues	[1 - 40] µm/s	Vitesse de nage des algues
Temps de division	[5 - 15] h	Temps entre deux division d'algues
Paramètres sur le stress		
Niveau de stress appliqué	[0 - 100] %	Niveau de stress appliqué aux algues une fois le seuil de déclenchement atteint Si le niveau de stress est à 100%, les cellules non agrégée sont détruites par l'environnement
Trigger	“Durée”	Déclencheur de stress Le stress sera appliqué à partir d'un certain temps écoulé
Seuil du trigger	Durée : [0 - 10] h	Seuil à partir duquel le stress est déclenché

Paramètres sur la simulation		
Longueur de la boîte (resp largeur)	[500-1000] mm	Dimensions de la boîte dans laquelle aura lieu la simulation
Temps de modélisation	[1 - 5] jours	Temps réel modélisé
Vitesse de simulation	{ "1 s / 1 min" ; "1 s / 10 min " ; "1 s / 1 h"}	Rapport entre le temps réel et le temps effectué dans la simulation Par exemple, 1s / 1min signifie que une seconde de vidéo correspond à 1min dans la modélisation

Tableau 3 : Paramètres pris en compte dans le GUI

Ce tableau présente tous les paramètres modifiables par l'utilisateur.rice du modèle avec leur explication

Nous avons dû produire en parallèle de la simulation un GUI en Python qui permet à l'utilisateur.rice de fixer des paramètres selon ses besoins. Cette interface, codée à l'aide de la bibliothèque PyQt5, nous a permis de proposer un modèle plus flexible qu'une modélisation dans laquelle l'utilisateur.rice ne pourrait avoir le choix de ces paramètres.

Dans un premier temps, nous avons pensé à réaliser une interface graphique qui aurait permis à l'utilisateur.rice d'interagir en direct avec la modélisation, avec notamment la possibilité de lancer le stress au moment choisi. Cependant, après s'être rendu compte du nombre de calculs nécessaire à la modélisation d'un grand nombre d'algues et de l'augmentation de ces calculs engendrée par l'ajout d'un stress, nous avons choisi de dissocier l'interface de l'affichage de la modélisation.

Ainsi l'interface finale sert d'introduction à la modélisation, et permet à l'utilisateur.rice de fixer les paramètres selon ses besoins avant de lancer la simulation. Vous trouverez en annexe une capture d'écran de l'interface ([Annexe 5](#))

Finalement, le modèle proposé par la simulation semble cohérent avec les observations faites, dès lors que l'on remarque une augmentation de la formation des agrégats en lien avec l'augmentation de l'adhésion des cellules.

2. Analyse d'image

Afin d'attester de la véracité de notre modèle, nous avons analysé les différentes données qui nous ont été fournis par Mr. Antoine DANON. Le but de cette analyse est, d'une part, de pouvoir avoir un aperçu du comportement des différentes cellules, et, d'autre part, de pouvoir comparer les résultats ressortant de l'analyse par rapport à la modélisation construite.

Premier pas, première méthode

Notre base de travail a été l'analyse de la vidéo transmise par notre tuteur, présentant la culture des algues vertes et leur réaction à l'application d'un stress oxydatif sur une période de plusieurs jours.

La première méthode que nous avions développé se basait sur le concept de niveau de gris représenté par un histogramme (Méthode : `methode_1_fail.py`). Dans cette méthode nous suivions un protocole consistant à extraire l'image et la transformer en différente valeur de gris, puis appliquer un effet de flou pour éliminer le bruit présent dans l'image créée. La suppression de bruit utilise le convolution, dans lequel nous appliquons à chaque élément une somme pondérée de ses voisins. De là, nous pouvons extraire un histogramme comportant le nombre de pixel par rapport à l'intensité de gris. Nous pouvons ensuite appliquer un masque qui assigne des valeurs à différents pixels de manière binaire (noir ou blanc) dépendant de la valeur trouvée. Enfin nous utilisons la fonction `skimage.measure.label` présente dans la bibliothèque `skimage` pour renvoyer le nombre d'éléments (pour plus de détails, voir [Annexe 4](#)).

Pour tester la limite de cette méthode nous avons observé comment cette méthode pouvait interpréter les contours par méthode de coloris.

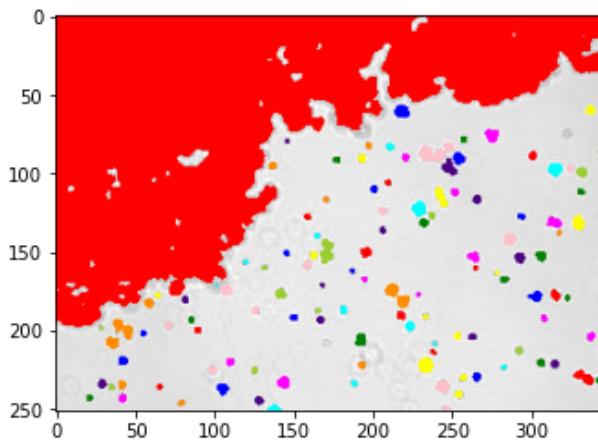


Figure 4 : Interprétation des contours par la méthodes de coloris

Les zones de couleurs représentent les contours repérés par l'algorithme de détection.

La Figure 4 nous présente immédiatement les limites de cette méthode car cette méthode ne nous donne pas de moyen de pouvoir différencier les cellules et les palmelloïdes.

Suite à cette observation, nous avons décidé de nous diriger vers une nouvelle bibliothèque Python destinée à l'analyse d'image.

Méthode d'analyse adoptée

Etant donné le problème rencontré dans la première méthode, nous avons cherché une autre manière de pouvoir analyser et cela avec les données pertinentes. Nous avons opté pour la bibliothèque *OpenCV*, qui est une bibliothèque spécialisée dans le traitement et l'analyse d'image.

Pareillement à la 1ère méthode, nous utilisons le même principe d'analyse mais avec ici deux choix de colorisation qui sont Gray et HSV ([Annexe 4](#)). Nous avons choisi la méthode HSV pour notre analyse. Ici, les deux choix se valent dès lors que les images sont noir et blanc.

Nous appliquons un thresholding sur cette images qui transforme en noir ou en blanc un pixel qui rentre dans la plage donné.

Suite à cela, nous utilisons différentes fonctions de la bibliothèque pour récupérer une matrice de convolution et l'appliquer sur l'image afin de supprimer les bruits. Enfin nous utilisons la fonction cv2.findContours sur l'image transformée pour obtenir un tableau contenant tous les contours trouvés. Nous créons alors un nouveau tableau contenant l'aire de ces contours prélevés et nous pouvons ainsi calculer le nombre de cellules.

Analyse de l'intégralité de la vidéo

Une fois les méthodes d'analyse d'image éprouvées, il nous est possible de faire en sorte d'analyser un grand nombre d'images pour observer des tendances dans le temps de l'évolution de la taille des agrégats ou du nombre de cellules.

Pour cela, nous avons tout d'abord extrait la vidéo en images, stockées dans un dossier images. Une vidéo de 39 secondes à 20 images par seconde représente donc environ 780 images à analyser. Ici, la vidéo a été prise sur 72h, soit 259 200 sec. Les images sont donc séparées d'un intervalle de temps de 5 minutes 30 secondes environ.

Nous avons ensuite créé un tableau image, représenté par tableau de matrices, que nous avons rempli avec l'ensemble des images contenues dans le dossier images. Ce tableau trié présente l'avantage de pouvoir appliquer facilement les algorithmes d'analyse. En effet, il nous a suffit d'implémenter les algorithmes d'analyse dans une boucle que l'on incrémente pour chaque image contenue dans le tableau image. Le tableau d'images étant trié de la première à la dernière image, il est ensuite possible d'observer les tendances et de tracer des graphiques pour une analyse plus aisée.

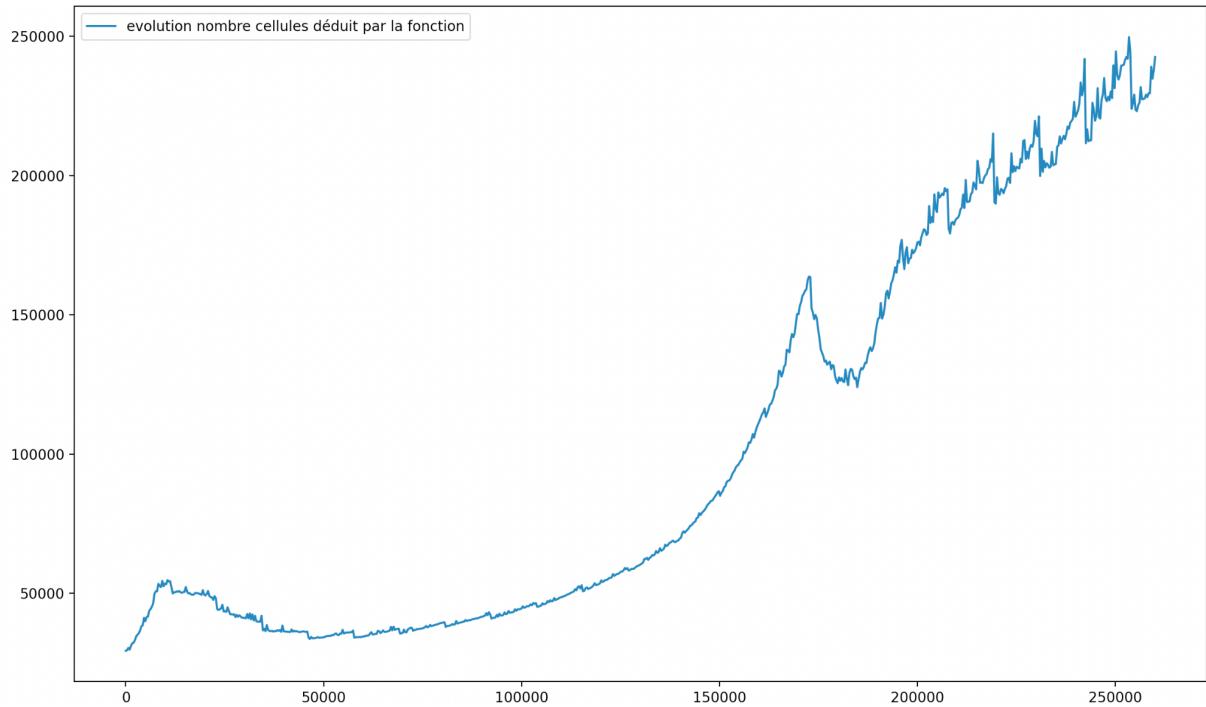


Figure 5 : Graphique du nombre de cellules en fonction du temps en secondes, déterminé par la 1e méthode

Le graphique de la Figure 5 semble cohérent, puisque le nombre de cellules augmente au fil du temps de manière exponentielle. Cependant l'analyse est nuancée car on observe un bruit important. De plus, le filtre réglé de telle sorte à mieux analyser l'image en présence de beaucoup de cellules, il est moins performant dans les 50000 premières secondes et renvoie des résultats moins pertinents.

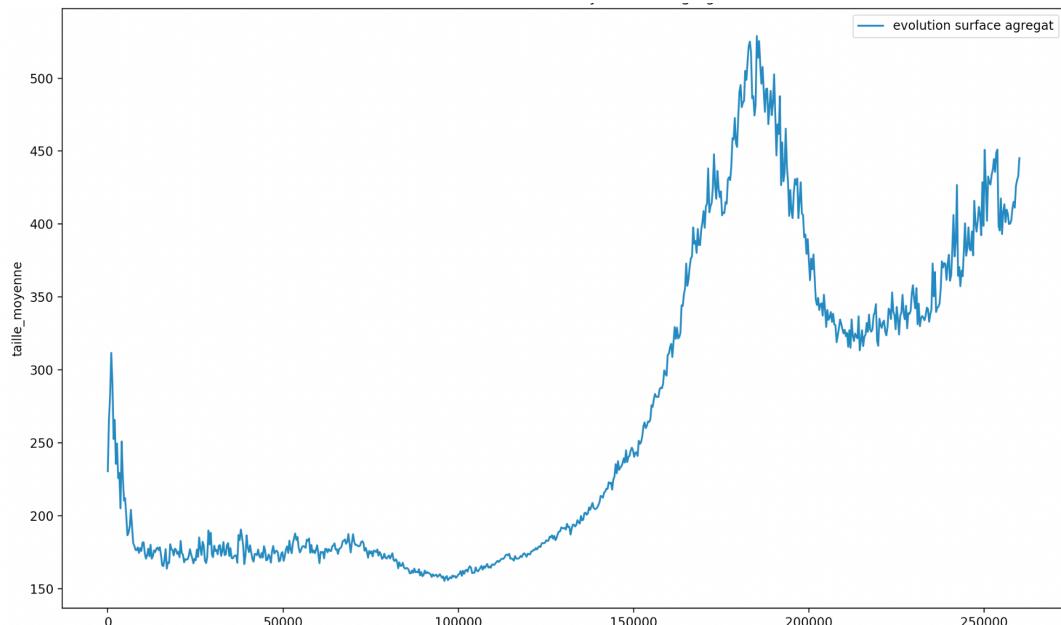


Figure 6 : Graphique de la taille moyenne des agrégats en pixels en fonction du temps en secondes

Ce graphique (Figure 6) confirme les limites de l'analyse en début de vidéo. Cependant, on observe bien une augmentation de la taille moyenne des agrégats. Tout d'abord les palmelloïdes se forment, et donc la moyenne de l'aire devrait logiquement augmenter. Ensuite les agrégats se forment, ce qui correspond à l'explosion de l'aire des agrégats. La chute importante au bout de 200 000s peut s'expliquer de trois manières, un problème d'algorithme de détection, le déplacement des agrégats hors du cadre de la vidéo, ainsi que les limites d'un modèle 2D lorsque les agrégats se "superposent".

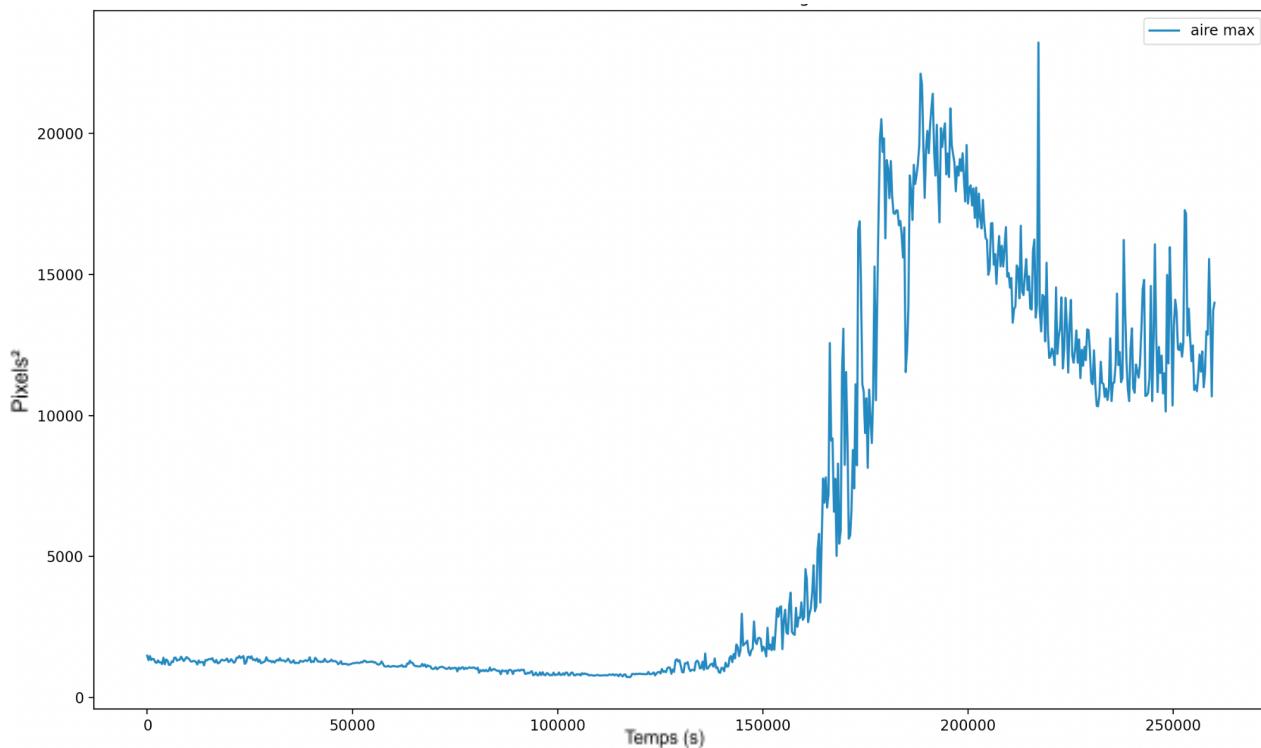


Figure 7 : Graphique de l'aire maximale des agrégats en pixels en fonction du temps en secondes

Nous avons développé deux méthodes de calculs pour compter le nombre de cellules. La première étant que nous prenons la plus petite cellule (caractérisée par la surface détectée) et on divise la somme des surfaces totales par la plus petite cellule.

Pour la deuxième méthode nous stockons dans une variable la taille de la plus petite cellule, on créer une variable qui correspond à la surface de 2 cellules pour signifier que deux cellules sont connectées. Avec ces variables nous allons parcourir le tableau de surface des contours déterminés précédemment, si la surface est plus grande que la surface de la plus petite cellule mais moins grande que la surface de 2 cellules on ajoute que 1 au compte de cellule. Cependant, si la surface est plus grande que la surface de 2 cellules nous procédons à une division euclidienne pour obtenir le nombre de cellules présentes dans la surface donnée. On ajoute ensuite au compte de cellule. (voir image dans Problèmes rencontrés)

Problèmes rencontrés

Extraction de la vidéo

L'extraction de chaque frame de la vidéo posait tout d'abord problème, en effet les images n'étant pas toutes extraites dans le bon fichier, ce problème a été résolu par la suite.

Tri du tableau d'images

L'analyse de la vidéo nécessitait ensuite d'appliquer l'algorithme d'analyse à l'intégralité des frames extraits de la vidéo en utilisant un tableau d'images. La difficulté ici était d'obtenir un tableau trié. La plupart des algorithmes donnaient un tableau aléatoirement construit. Le problème fut ensuite résolu.

Tache de lumière

Ensuite, il nous est apparu qu'une tache de lumière sûrement due à un reflet du soleil grossissait au fil des images, faussant tous les résultats de l'analyse, il nous fallut donc redimensionner les images, et donc perdre de l'information. Environ un cinquième du bas de chaque image a été coupé.

Nombre trop important d'algues

Sur les analyses des "frames" de fin de vidéo, il nous est apparu des résultats parfois incohérents, l'algorithme de comptage des algues n'arrivant plus forcément à les distinguer clairement, il fallut alors ajuster le filtre hsv appliqué aux images pour leur analyse. Pour illustrer cela voici un graphique correspondant à l'évolution du nombre de cellules en fonction des images.

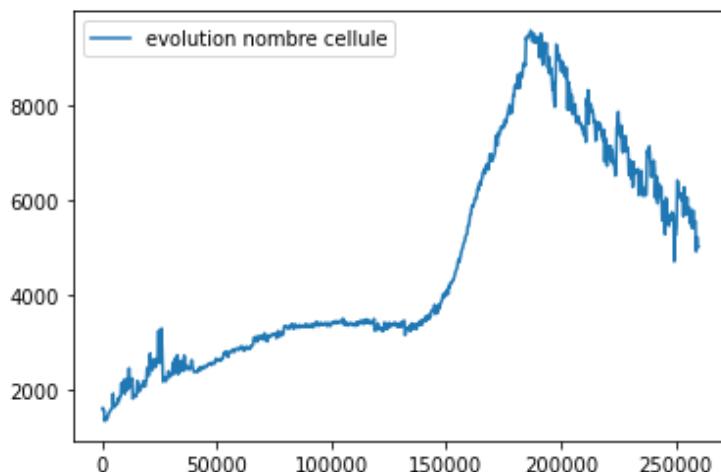


Figure 8 : Graphique du nombre de cellules en fonction du temps en secondes, méthode peu concluante

Cette incohérence peut aussi être expliquée par le fait qu'il faille adapter le masque car les conditions de luminosité sont légèrement altérées au cours de l'expérience (observé par la présence de l'anneau de lumière en bas d'image).

Conclusion

Le résultat de ce projet est un modèle concluant, qui possède des bases cohérentes avec les observations. En effet, la modélisation des *Chlamydomonas reinhardtii* est semblable aux données que nous a transmises notre encadrant, et le paramètre de l'adhérence des cellules semble bel et bien avoir un rôle dans la formation des agrégats.

Nous gardons en piste d'amélioration le fait que, pour l'instant, nous ne savons pas qui des palmelloïdes ou des cellules initient la formation d'agrégats. Nous avons donc dû poser l'hypothèse que les deux éléments sont capables d'un tel comportement, hypothèse qui reste à vérifier. De plus, la récence des recherches a entraîné un manque d'informations quant aux différentes lois de réaction des algues face au stress, et à ainsi engendré de nouvelles hypothèses qui réduisent la précision de notre modèle.

Annexes

1. Code du projet

GitHub du projet : https://github.com/Nicolassalvan/Projet_Aggregation (Annexe 1)

Le fichier README du dossier permet de comprendre le fonctionnement du code

Le dossier EXE contient un fichier .exe permettant une plus simple utilisation de l'interface

2. Sources

- ❖ *Chlamydomonas reinhardtii* (Annexe 2.1)
 - https://en.wikipedia.org/wiki/Chlamydomonas_reinhardtii (Wikipédia, 2022)
 - <https://www.ibps.sorbonne-universite.fr/fr/Recherche/umr-7238/biologie-synthetique-systemique-microalgues> (IBPS, 2022)
 - <https://www.gettyimages.fr/vid%C3%A9os/chlamydomonas> , Image de la page de garde
 - <https://www.universalis.fr/dictionnaire/palmelloide/> (Universalis, 2023)
- ❖ Forums (Annexe 2.2)
 - <https://stackoverflow.com/> (Stack Overflow, 2022)
- ❖ Documentation (Annexe 2.3)
 - <https://pypi.org/project/PyQt5/> (PyPi, 2022)
 - <https://www.riverbankcomputing.com/static/Docs/PyQt5/> (RiverBank, 2022)
 - <https://matplotlib.org/> (Matplotlib, 2022)
 - <https://numpy.org/> (Numpy, 2022)

3. Glossaire

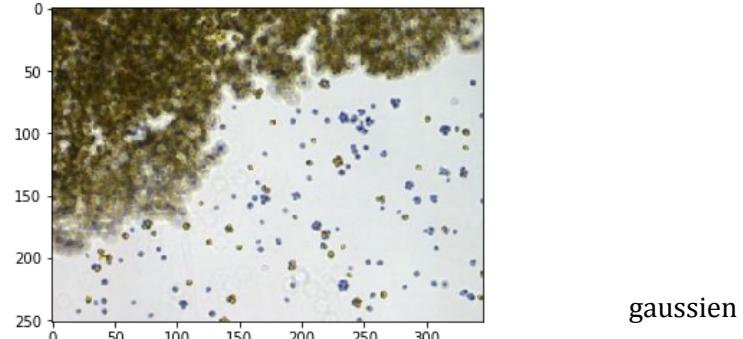
- ❖ Palmelloïde
 - Formation cellulaire, induite par un stress abiotique dans le milieu de culture, composée de cellules se divisant sans se séparer, contenues par une gaine mucilagineuse
- ❖ Agrégat
 - Schéma d'organisation cellulaire dans lequel les cellules s'agrègent en se collant les unes aux autres, leur permettant de survivre au stress.
- ❖ Stress abiotique
 - Changement de condition de culture, que ce soit dans la température, l'exposition ou la composition du milieu de culture.

4. Analyse d'image

Détail et graph obtenu par la première méthode

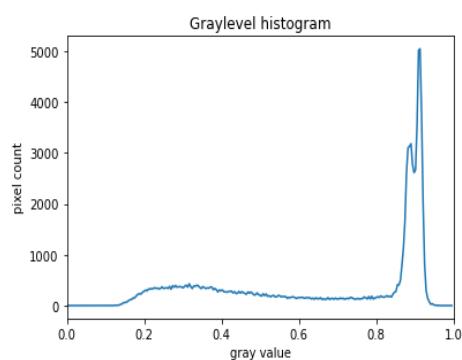
Annexe .1 : image de référence

Annexe 4.2 : Gris et Flou

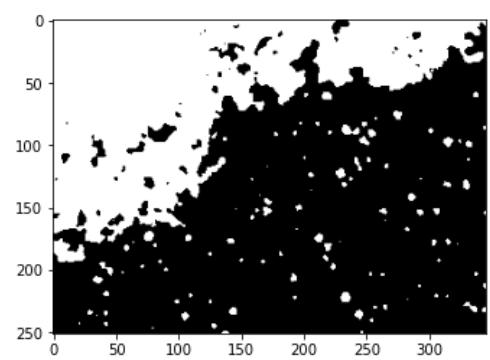


gaussien

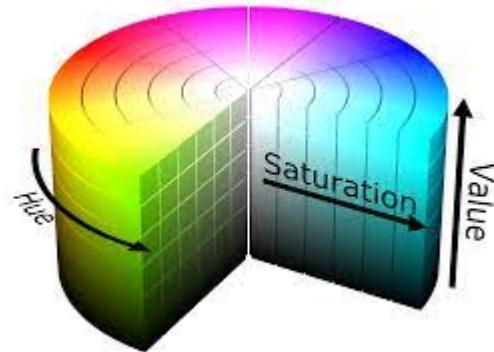
Annexe 4.3 : Histogramme des niveaux de gris pour image flou gaussien



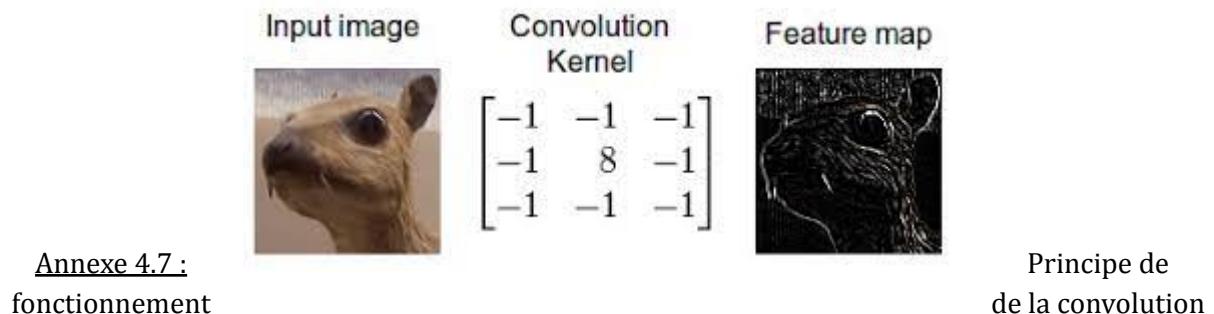
Annexe 4.4 : Image résultante de l'application du masque (<0.75)



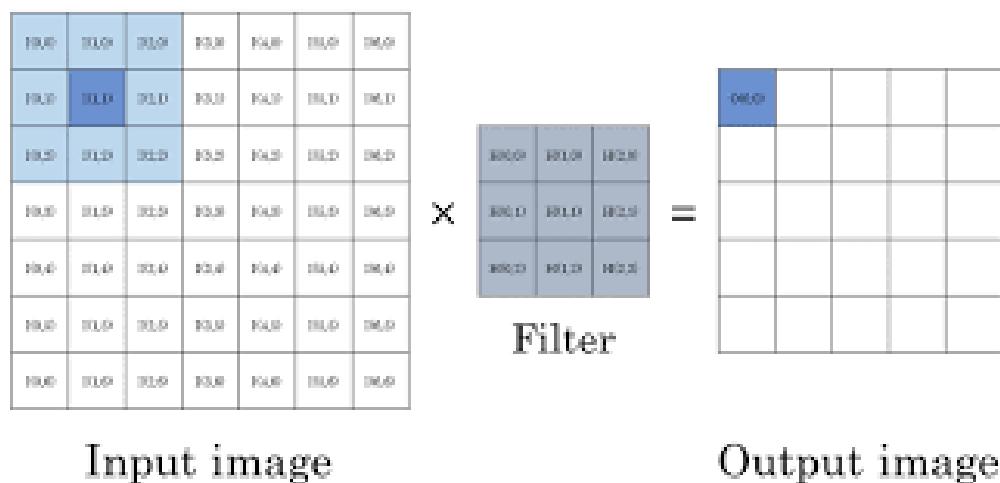
Annexe 4.5 : principe de couleurs HSV



Annexe 4.6 : Utilisation de la méthode de convolution pour déterminer les contours.



Annexe 4.7 :
fonctionnement



Annexe 4.8 : Explication de la fonction `skimage.measure.label()`

label

`skimage.measure.label(Label_image, background=None, return_num=False, connectivity=None)`

[source]

Label connected regions of an integer array.

Two pixels are connected when they are neighbors and have the same value. In 2D, they can be neighbors either in a 1- or 2-connected sense. The value refers to the maximum number of orthogonal hops to consider a pixel/voxel a neighbor:

1-connectivity	2-connectivity	diagonal connection close-up
[]	[] [] []	[]
	\ /	<- hop 2
[]-[x]-[]	[]-[x]-[]	[x]-[]
	/ \	hop 1
[]	[] [] []	

5. Interface graphique

Annexe 5.1 : Vue globale de l'interface

The screenshot shows a software window titled "Modélisation Chlamydomonas Reinhardtii". The interface is divided into several sections:

- Bienvenue**:
 - Bienvenue, Dans cette interface, vous avez à déterminer les différents paramètres qui seront pris en compte pour la modélisation
 - Ici, le stress sera abiotique (thermique, hydrique, oxydatif, lumineux...). Dans ce modèle, on associe le stress à une probabilité de survie de la cellule et à une architecture en agrégats des cellules Afin de faciliter les choix possibles, le stress sera désigné en tant que pourcentage. Dépassé 100, les cellules sont détruites par l'environnement.
 - Bien à vous.
- Données de la modélisation**:
 - Surface de la boîte : 25 mm²
 - Densité d'algues : 0.8 /cm²
 - Diamètre d'une algue : 2 µm
- Simulation**:
 - Voulez-vous appliquer un stress lors de la modélisation ? Oui Non
 - Lancer la simulation : gif
 - Lancer la simulation : mp4
- Paramètres de l'algue**:
 - Vitesse moyenne : 10 m/s
 - Nombre d'algues initiale : 20
 - Temps de division : 10 h
- Paramètres du stress**:
 - Niveau de stress : 50 %
 - Choix du déclencheur de stress : Durée
 - Seuil de déclenchement du stress : 10 x 10³ algues
- Paramètres de la simulation**:
 - Longueur de la boîte : 5 mm
 - Largeur de la boîte : 5 mm
 - Temps de modélisation : 0 j
 - Vitesse de la simulation : 0 h/s

Annexe 5.2 : Message explicatif et constantes de la modélisation

The screenshot shows a software window titled "Modélisation Chlamydomonas Reinhardtii". The interface is divided into several sections:

- Bienvenue**:
 - Bienvenue, Dans cette interface, vous avez à déterminer les différents paramètres qui seront pris en compte pour la modélisation
 - Ici, le stress sera abiotique (thermique, hydrique, oxydatif, lumineux...). Dans ce modèle, on associe le stress à une probabilité de survie de la cellule et à une architecture en agrégats des cellules Afin de faciliter les choix possibles, le stress sera désigné en tant que pourcentage. Dépassé 100, les cellules sont détruites par l'environnement.
 - Bien à vous.
- Données de la modélisation**:
 - Surface de la boîte : 25 mm²
 - Densité d'algues : 0.8 /cm²
 - Diamètre d'une algue : 2 µm

Annexe 5.3 : Lancement de la simulation

Simulation

Voulez-vous appliquer un stress lors de la modélisation ? Oui Non

Annexe 5.4 : Paramètres de la simulation

Paramètres de la simulation

Longueur de la boite :

Largeur de la boite :

Temps de modélisation :

Vitesse de la simulation :