À LA RECHERCHE DU CRABE BLEU

Projet porté Jade BAHI, Luana GANCI, Khallil KADRI, Emma MONTERO, Louane PERO, Jasmine SAÏGHI et Laura TESSIER et encadré par M. GUIGUE

Vidéo du projet : https://youtu.be/3zW4QEFpXQo
Blog du projet : https://explorelabz.org/dnl-artaud-2025-2026-le-blog/

Comment détecter automatiquement la présence de larves de crabe bleu dans l'étang de Berre à l'aide de la microscopie et de l'intelligence artificielle ?

Lycée Antonin Artaud
Marseille (13)

Rapport de projet
Concours CGénial 2026

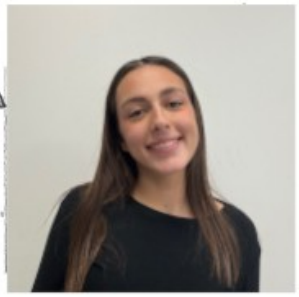
En partenariat avec
Fairscope
8 Vies pour la planète

Présentation de l'équipe

Notre groupe est composé de sept élèves de Première (filières Général et ST2S) du lycée Antonin Artaud à Marseille. Nous avons travaillé ensemble depuis le début d'année sur ce projet. Nous nous sommes réparties les tâches selon nos goûts et nos compétences, mais une majorité du travail a été réalisé par l'ensemble de l'équipe !

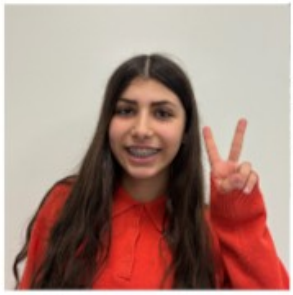
Louane
Cheffe de projet

Luana
Future danseuse et psychologue du groupe.



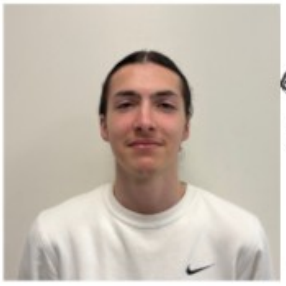
Emma
Féministe en herbe

Jade
Le pont entre les sciences et l'anglais



Jasmine
Rédactrice du Blog

Laura
Fan de son cheval : Magnetic



Khalil
Passionné de cuisine

Sommaire

I. Introduction

1.1 Le plancton : des êtres minuscules, un rôle immense

Le plancton désigne l'ensemble des organismes microscopiques qui vivent en suspension dans l'eau, que ce soit en mer ou en eau douce. Ces êtres vivants sont incapables de nager à contre-courant et se laissent porter par les eaux, parfois jusqu'à 200 mètres de profondeur. Malgré leur taille infime — souvent inférieure au millimètre — ils jouent un rôle fondamental dans les écosystèmes aquatiques.

Le plancton se divise en deux grandes familles : le phytoplancton, composé d'organismes végétaux (algues microscopiques, cyanobactéries…) qui réalisent la photosynthèse et produisent une grande partie de l'oxygène de notre planète, et le zooplancton, constitué d'organismes animaux (petits crustacés, larves de poissons, méduses…) qui se nourrissent du phytoplancton et constituent la base de la chaîne alimentaire marine.

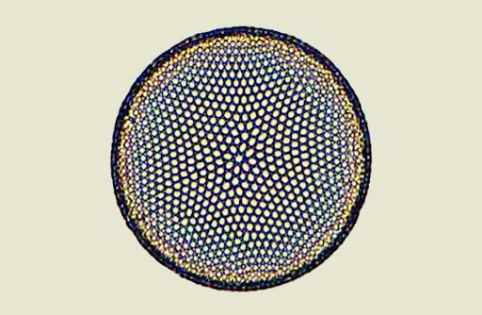


Fig. 1a — Phytoplancton : exemple de diatomée

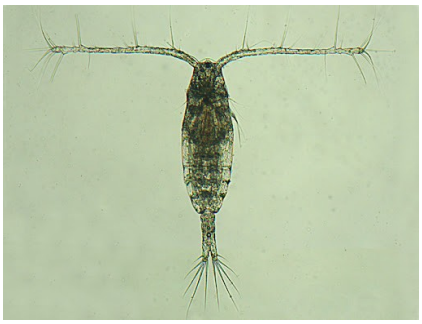


Fig. 1b — Zooplancton : exemple de copépode

Notre projet s'intéresse uniquement au zooplancton, et plus précisément à une larve particulière : la zoée du crabe bleu.

Qu'est-ce que la zoée ?

La zoée est le stade larvaire des crabes, y compris du crabe bleu. À ce stade, la larve est microscopique, transparente et difficile à distinguer à l'œil nu. Elle dérive dans les eaux pendant plusieurs semaines avant de se transformer en mégalope, puis en juvénile.

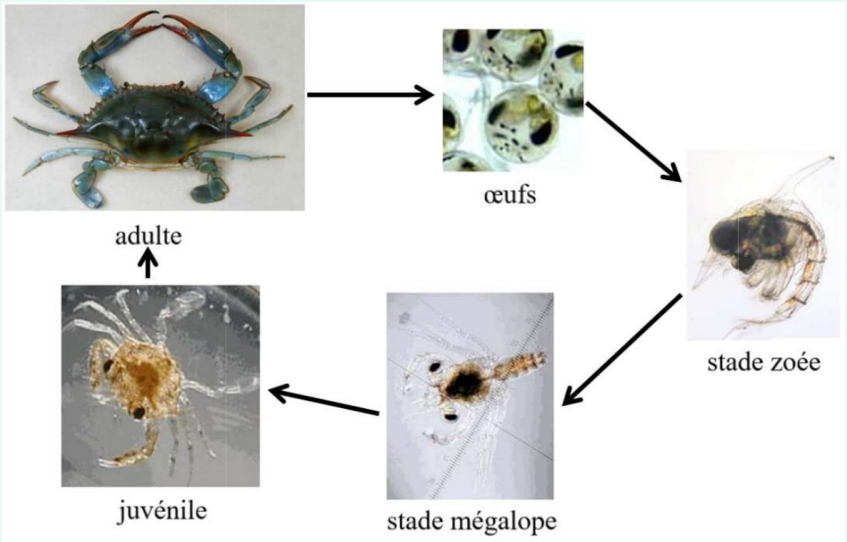


Fig. 2 — Cycle de vie du crabe bleu, Office de l'Environnement de la Corse

1.2 Le crabe bleu : une espèce invasive menaçante

Le crabe bleu (*Callinectes sapidus*) est un crustacé décapode originaire des côtes atlantiques d'Amérique du Nord. Son nom vient du latin : « callinectes » signifie « beau nageur » et « sapidus » veut dire « savoureux ». En Amérique, il est d'ailleurs un mets très apprécié et fait l'objet d'une pêche importante.

Son introduction en Méditerranée, vraisemblablement via les eaux de ballast des navires, a provoqué une véritable catastrophe écologique. Très résistant et extrêmement vorace, il se reproduit en grand nombre et n'a dans ces eaux pratiquement aucun prédateur naturel. Il colonise très rapidement les milieux où il s'installe.

Fig. 3a — Le crabe bleu adulte (Callinectes sapidus)

Fig. 3b — Zoée du crabe bleu, stade précoce

L'étang de Berre, situé à quelques kilomètres de Marseille et faisant 155 km², est le plus grand étang saumâtre d'Europe occidentale. Ses eaux calmes, peu profondes et riches en nourriture constituent un environnement idéal pour le développement du crabe bleu. Sa prolifération dans l'étang représente une menace directe pour :

- la biodiversité locale : le crabe bleu s'attaque aux palourdes, aux moules, aux petits poissons et aux invertébrés ;
- les activités de pêche : les filets sont régulièrement endommagés et les stocks de poissons réduits, entraînant de lourdes pertes économiques ;
- l'équilibre de l'écosystème : en perturbant la chaîne alimentaire, il déstabilise tout l'écosystème de l'étang.

1.3 Notre démarche pour répondre au problème

Pour suivre l'évolution de la population de crabe bleu dans l'étang de Berre, il est essentiel de comprendre son cycle de reproduction. La clé se trouve à la phase larvaire : si l'on peut détecter la présence de zoées dans les eaux de l'étang au printemps (période de reproduction), on peut estimer les futurs effectifs et identifier les zones de ponte.

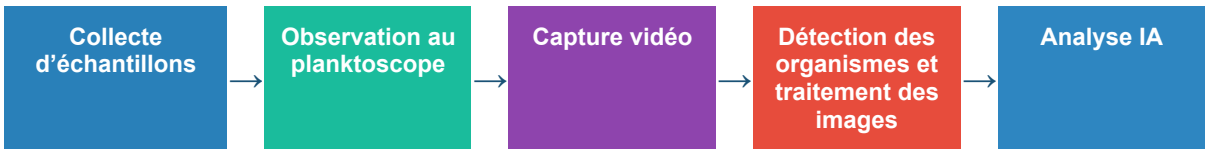
L'étude du plancton se fait traditionnellement par analyse de l'ADN environnemental (eDNA), une méthode puissante mais coûteuse et longue, qui nécessite des connaissances en biologie marine avancées. Nous avons choisi une approche plus accessible et originale : la microscopie couplée à l'intelligence artificielle.

Problématique :

Comment détecter automatiquement la présence de larves de crabe bleu dans l'étang de Berre à l'aide de la microscopie et de l'intelligence artificielle ?

Notre démarche repose sur trois grandes étapes :

- Construire un microscope à plancton open-source d'après le planktoscope de Fairscope pour observer les échantillons prélevés ;
- Développer une chaîne de traitement Python capable de détecter automatiquement les organismes qui passent devant la caméra, d'en capturer l'image et de la traiter (isolation de l'organisme sur fond blanc)
- Entraîner une intelligence artificielle à reconnaître la zoée du crabe bleu parmi tous les autres organismes présents dans le prélèvement, afin de détecter sa présence et de la quantifier.



Protocole de mesure complet

1.4 Présentation des partenaires

Fairscope

Fairscope est une start-up française fondée en 2023 par Thibaut Pollina et David Le Guen, basée à Morlaix (Bretagne). Son équipe de 7 personnes conçoit des instruments scientifiques open-source dédiés à la biologie marine, avec pour cœur de métier le planktoscope : un microscope à flux continu permettant l'imagerie des communautés planctoniques. Proposé à moins de 5 000 euros (environ vingt fois moins cher que ses équivalents industriels) il a déjà été vendu à plus de 110 laboratoires dans 26 pays. Fairscope participe à EcoTaxa, la plateforme mondiale de classification du plancton, sur laquelle près de 50 millions d'images issues du planktoscope ont été déposées. Pour notre projet, Fairscope nous a fourni les pièces les plus difficiles à se procurer (structure bambou découpée au laser, HAT Raspberry Pi, capillaires en verre), une liste de matériel et un cahier des charges, et nous a guidés tout au long de l'assemblage et de la configuration logicielle.

8 Vies pour la planète

8 Vies pour la planète est une association fondée en 2018, basée à Saint-Chamas, au bord de l'étang de Berre. Sa mission est de promouvoir l'innovation au service de l'environnement, en combinant projets techniques, artistiques et participatifs. Parmi ses actions phares : le projet ZoRRO (réimplantation d'herbiers de zostères, labellés par la Décennie de l'ONU pour la restauration des écosystèmes), des ramassages de déchets en milieu aquatique, des mesures citoyennes de la qualité de l'eau et de l'air, et des ateliers pédagogiques gratuits dans les écoles. L'association dispose d'un bateau-laboratoire, le batOlab, pour ses actions sur l'étang. Elle nous a apporté un soutien financier pour l'achat du matériel, nous a sensibilisés aux enjeux écologiques de l'étang de Berre, et nous accompagnera lors des prélèvements de printemps. Ce projet est important pour 8 Vies, qui aimerait ensuite utiliser notre protocole de manière régulière afin de suivre la population de crabe bleu dans l'étang de Berre. L'avantage est que le microscope est peu coûteux, et que la démarche ne nécessite pas des connaissances trop poussées en biologie marine : elle peut donc être suivie par n'importe quel citoyen engagé.

II. Le Planktoscope

2.1 Qu'est-ce que le planktoscope ?

Le planktoscope est un microscope spécialement conçu pour observer le plancton. Contrairement aux microscopes de laboratoire traditionnels, il ne sert pas à observer un échantillon immobile sur une lame : il fait défiler le liquide contenant les organismes devant la caméra grâce à un système microfluidique. Il est ainsi possible de « scanner » automatiquement un grand volume d'eau et de photographier les organismes les uns après les autres.

C'est un outil entièrement open-source : ses plans de fabrication sont disponibles en ligne et chacun peut le construire. Cela correspond parfaitement à notre démarche : utiliser des technologies accessibles pour faire de la science participative.

Le planktoscope v2.6 que nous avons fabriqué repose sur un Raspberry Pi (un micro-ordinateur de la taille d'une carte de crédit) qui contrôle l'ensemble du système (caméra, pompe, moteurs) et crée un réseau Wi-Fi local permettant de piloter l'appareil depuis un ordinateur ou une tablette via une interface web fournie par Fairscope.

2.2 Construction de l'appareil

La fabrication du planktoscope a été l'une des étapes les plus enrichissantes de notre projet. Fairscope nous a fourni les éléments spécifiques et difficiles à se procurer : la structure découpée au laser dans du bois de bambou, le HAT (carte d'extension développée par Fairscope) dédié au Raspberry Pi pour contrôler les moteurs et la pompe, ainsi que les capillaires en verre de 300 µm. Le reste des composants (Raspberry Pi, caméra, lentilles, moteurs pas à pas, pompe, visserie et connectique) a été acheté par nos soins selon le cahier des charges fourni par Fairscope, en l'adaptant à notre budget.

Étape	Description
1	Achat et inventaire des pièces : le kit contient plus d'une centaine de composants (pièces en bambou, vis, moteurs, lentilles, capillaire, seringue, Raspberry Pi, caméra, HAT, LED…). Nous avons commencé par les identifier et les trier.
2	Assemblage de la structure en bambou : les pièces découpées au laser s'emboîtent et se vissent pour former le châssis. Cette étape demande de la précision car les tolérances sont faibles.
3	Installation et branchement des moteurs pas à pas : deux moteurs permettent de déplacer le porte-échantillon avec une précision micrométrique pour ajuster la mise au point.
4	Montage du système optique : les deux lentilles convergentes doivent être placées de manière précise à l'intérieur d'un tube optique. La lentille oculaire doit se situer à 25,00 mm du capteur. Si cette distance n'est pas précisément ajustée, l'image ne sera pas nette.
5	Câblage électronique : le Raspberry Pi, le HAT, les moteurs, la pompe et les LEDs sont connectés et soudés ensemble selon le schéma fourni dans la documentation Fairscope.
6	Branchement du chemin microfluidique et vérification de son étanchéité.
7	Configuration logicielle : une fois assemblé, le Raspberry Pi est configuré pour démarrer automatiquement l'interface web du planktoscope.

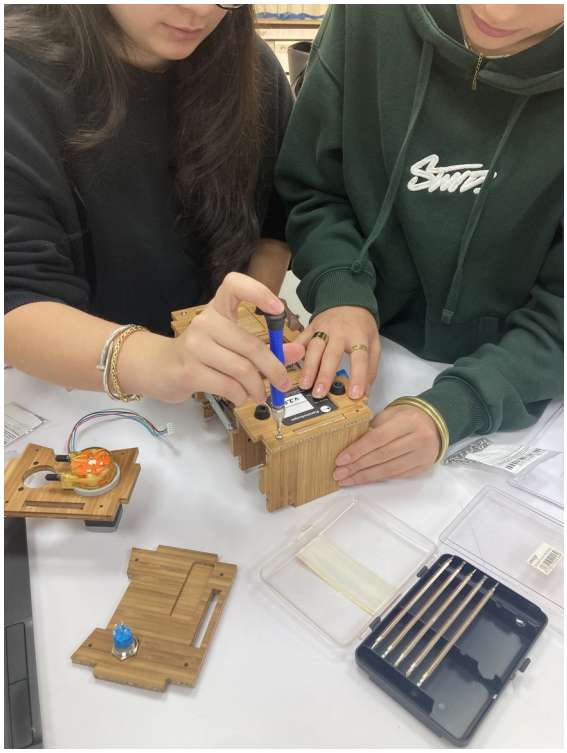


Fig. 4a — Assemblage de la structure

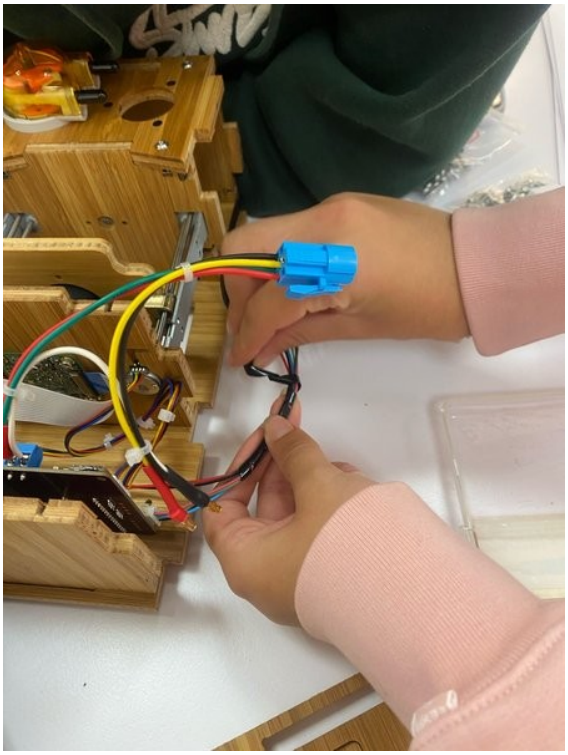


Fig. 4b — Câblage de l'interrupteur, de la LED et de la pompe

Fig. 4c — Fixation des moteurs et du microscope

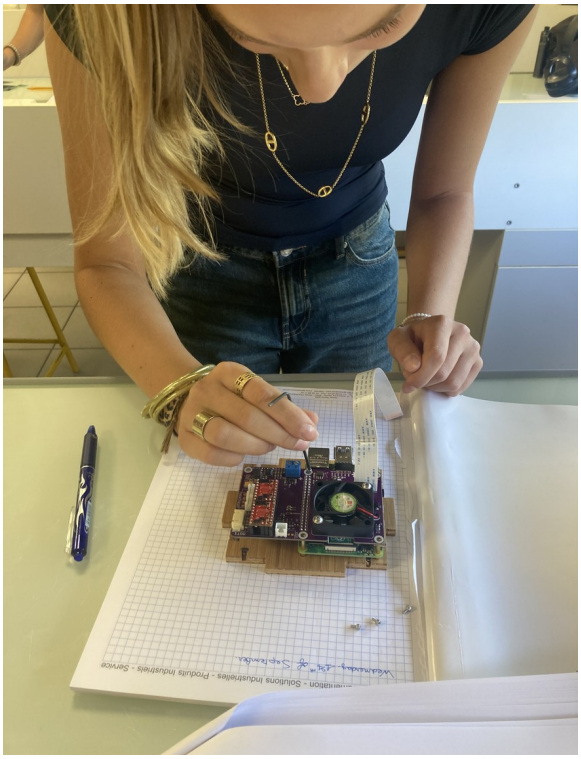


Fig. 4d — Fixation du HAT et du raspberry PI

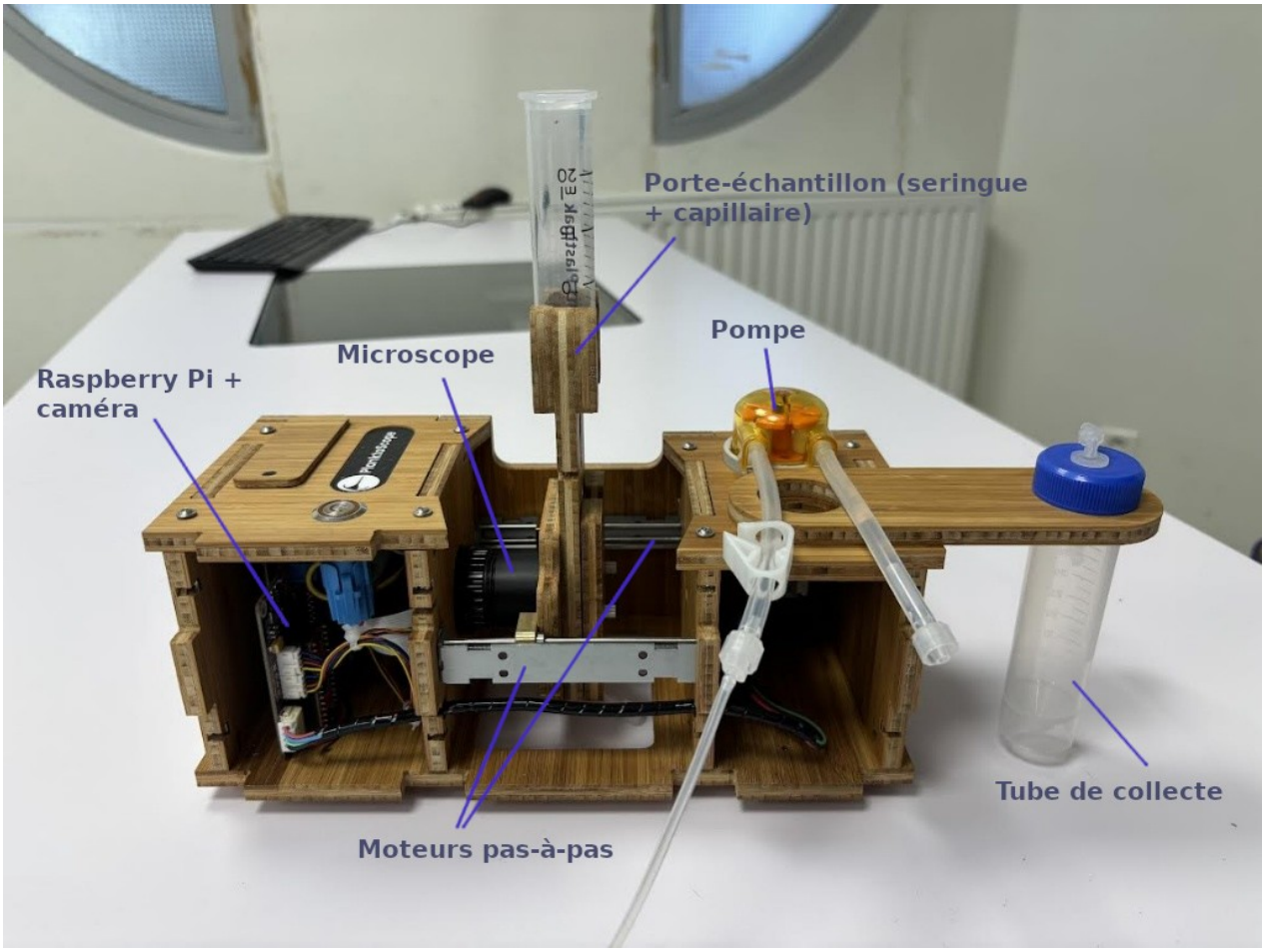


Fig. 5 — Le planktoscope assemblé

2.3 Le circuit microfluidique

L'originalité du planktoscope réside dans son circuit microfluidique, permettant d'analyser beaucoup plus d'eau qu'avec un microscope classique. Au lieu de déposer manuellement chaque organisme sous le microscope, on fait défiler toute la solution à analyser à travers un capillaire transparent de 300 µm de diamètre intérieur. Ce capillaire est positionné exactement au foyer de la lentille objectif. La figure 6 résume le parcours de l'échantillon dans le circuit microfluidique de notre microscope.

Fig. 6 — Schéma du circuit microfluidique du planktoscope

La pompe péristaltique aspire lentement le liquide depuis la seringue. Son débit est réglable depuis l'interface de l'ordinateur, ce qui permet d'espacer les planctons pour éviter qu'ils ne se superposent devant la caméra, et de faire en sorte que l'image ne soit pas floue à cause d'une vitesse trop rapide par rapport à fréquence de capture de la caméra. L'échantillon passe dans le capillaire et est observé par la caméra à travers le microscope, puis continue son chemin à travers la pompe. Enfin, il est récupéré dans un tube de collecte.

2.4 L'optique : deux lentilles convergentes

Le système optique du planktoscope est un microscope formé de deux lentilles convergentes. Le principe est similaire à celui d'un microscope classique de lycée, mais ici l'image est projetée sur le capteur au lieu d'être renvoyée à l'infini.

- La lentille objectif (distance focale f_1 = 12 mm) est placée devant le capillaire. Le plancton que l'on souhaite observer se trouve au foyer objet de cette lentille, on ajuste la position grâce aux moteurs permettant de déplacer le capillaire. En effet, comme il fait 300 µm d'épaisseur, le plancton n'est pas forcément exactement au plan focal objet. Pouvoir actionner les moteurs et déplacer le porte-échantillon permet de corriger cela.
- La lentille oculaire (distance focale f_2 = 25 mm) projette l'image finale sur le capteur. Elle doit se situer précisémment à 25,00 mm du capteur, car cette distance ne peut pas être ajustée ensuite. Nous avons donc dû visser les lentilles dans un tube optique et les positionner dedans au pied à coulisse.
- La caméra Raspberry Pi (capteur IMX477) capture l'image finale.

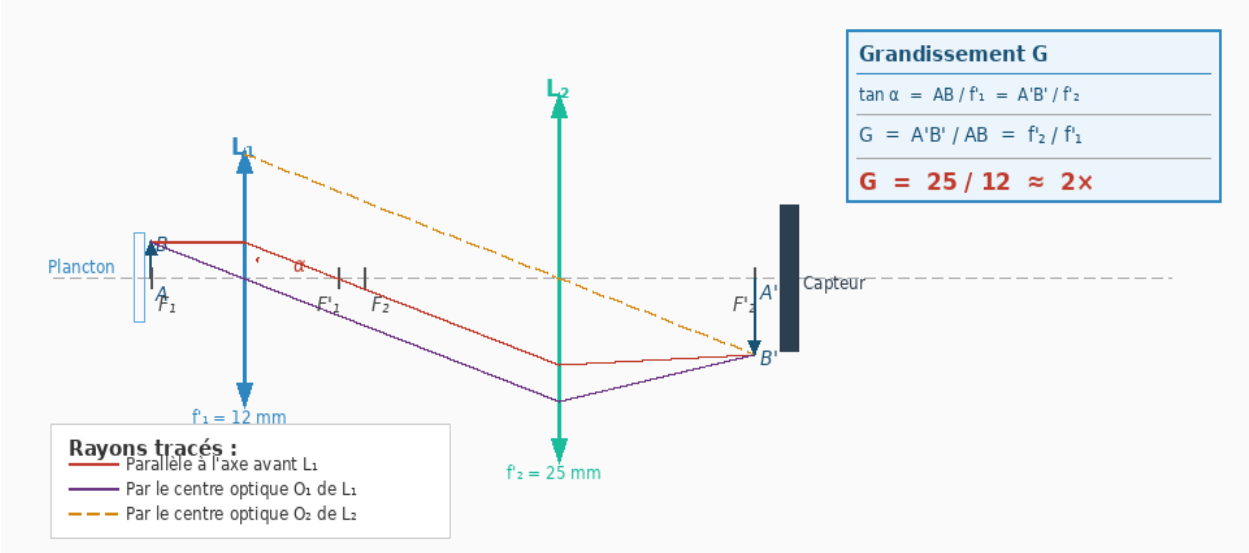


Fig. 7 — Schéma d'optique géométrique du système à deux lentilles convergentes

Démonstration et calcul du grandissement

L'objet AB (le plancton dans le capillaire) est placé dans le plan focal objet F_1 de L_1, c'est-à-dire à la distance f'_1 = 12 mm de la lentille objectif. Dans ces conditions, les rayons issus d'un même point de l'objet ressortent de L_1 en faisceau parallèle, incliné d'un angle α par rapport à l'axe optique.

Un rayon issu de B passant par le centre optique O_1 de L_1 n'est pas dévié. Il repart avec le même angle α tel que :

$$\tan \alpha = AB / f'_1$$

Ce faisceau parallèle (incliné de α) entre ensuite dans la lentille oculaire L_2 (f'_2 = 25 mm). La distance entre L_1 et L_2 n'est pas importante. Un faisceau parallèle incliné de α converge dans le plan focal image F'_2 de L_2, là où est placé le capteur de la caméra, à une hauteur A'B' telle que :

$$\tan \alpha = A'B' / f'_2$$

Les deux expressions de tan α sont donc égales :

$$AB / f'_1 = A'B' / f'_2$$

Ainsi, on peut calculer le grandissement γ de notre microscope :

$$\gamma = A'B' / AB = f'_2 / f'_1 = 25 / 12 \approx 2$$

Un grandissement de 2 signifie qu'une zoée de 200 µm dans le capillaire produit une image de 400 µm sur le capteur. Avec un pixel de 1,55 µm pour la Pi Camera IMX477, cela représente environ 258 pixels de longueur : c'est une taille largement suffisante pour distinguer la morphologie caractéristique de la zoée (rostre, épines, segments abdominaux).

Taille réelle de la zoée	150 à 250 µm (stades précoces)
Taille sur le capteur (×2)	300 à 500 µm
Taille en pixels (1 px = 1,55 µm)	193 à 323 pixels
Résolution capteur IMX477	4056 × 3040 pixels

Lorsque l'image capturée est affichée sur un écran d'ordinateur classique (1920 × 1080 px ou plus), elle est zoomée numériquement. Un objet de 200 pixels sur le capteur occupe alors plusieurs centimètres à l'écran. Le grandissement optique ×2 est donc complété par un agrandissement numérique à l'affichage, et c'est cette combinaison qui rend les organismes observables et identifiables. Pour notre IA, c'est la taille en pixels qui compte : avec 200 pixels de longueur, la zoée présente suffisamment de détails morphologiques pour être reconnue.

2.5 L'acquisition d'images

Une fois le planktoscope allumé, il crée un réseau Wi-Fi local. En s'y connectant depuis un ordinateur et en ouvrant l'interface web fournie par Fairscope, on peut contrôler tous les paramètres de l'acquisition : débit de la pompe, position du porte-échantillon (mise au point via les moteurs pas à pas), paramètres de la caméra (exposition, gain), et déclencher la capture.

Le flux vidéo est enregistré sur la carte microSD du Raspberry Pi, ou peut être capturé directement sur l'ordinateur, avant d'être traité par nos scripts Python. Le débit doit rester suffisamment faible pour éviter le flou de mouvement et la superposition des organismes.

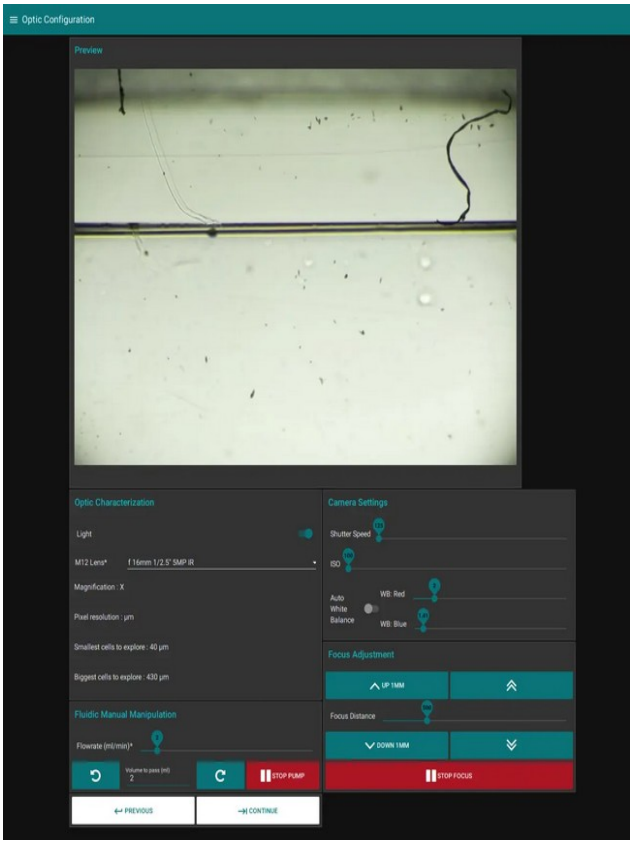


Fig. 8 — Capture d'écran de l'interface web du planktoscope

III. L'Intelligence Artificielle avec Teachable Machine

3.1 Présentation de Teachable Machine

Teachable Machine est un outil en ligne gratuit développé par Google qui permet de créer facilement des modèles d'intelligence artificielle (IA) sans avoir besoin de programmer. Il est particulièrement adapté aux débutants et aux projets éducatifs.

Le principe repose sur l'apprentissage automatique (machine learning) : au lieu d'écrire des règles à la main pour identifier un objet, on montre à l'ordinateur de nombreux exemples de ce qu'on veut reconnaître, et il apprend lui-même à le détecter. Plus on lui fournit d'exemples variés, plus il sera fiable.

Comment fonctionne l'entraînement ?

On crée des « classes » (catégories), on charge des photos dans chaque classe, puis on clique sur « Entraîner ». Teachable Machine analyse les différences visuelles entre les classes et construit un modèle capable de classer de nouvelles images. Ce modèle peut ensuite être exporté au format TensorFlow/Keras (.h5) et utilisé dans un programme Python.

3.2 Constitution du jeu de données d'entraînement

Avant de travailler sur les images de plancton, nous avons d'abord réalisé un test simple en entraînant l'IA à distinguer des photos de chiens de photos de chats. Une fois ce test concluant, nous avons pu passer à notre vrai sujet.

Pour notre projet, nous avons constitué deux classes à partir d'images trouvées dans des banques d'images scientifiques en ligne :

- Classe 1 — Zoée du crabe bleu : images de larves de crabe bleu au stade zoée
- Classe 2 — Autre plancton : images d'organismes variés (copépodes, diatomées, rotifères, nauplies…) pour que l'IA apprenne à distinguer la zoée de tout le reste.

La sélection des images d'entraînement est une étape déterminante. En testant différentes configurations, nous avons identifié ce qui fonctionne bien et ce qui pose problème :

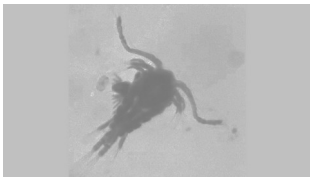
Critère	Image exemple	Explication	Notre choix
Fond de l'image		Le planktoscope produit des images sur fond clair (beige/blanc). Il faut entraîner l'IA sur des images aux mêmes conditions.	**Images sur fond clair uniquement**
Autres éléments sur l'image		Une image doit correspondre à un plancton seul, et il n'y doit pas y avoir de texte sur l'image pour ne pas tromper l'IA.	**Images de planctons seuls**
Netteté		Une image floue ne donne pas suffisamment de détails morphologiques à l'IA pour apprendre.	**Images nettes uniquement**
Stade larvaire		La morphologie change selon le stade. Un stade absent du jeu de données risque de ne pas être reconnu.	**Stades précoces**

Tableau 1 — Critères de sélection des images d'entraînement

Nous avons entraîné notre IA avec 80 images de planctons prises selon les critères ci-dessus, et 78 images de zoée du crabe bleu. Ce jeu de données reste limité. Les modèles utilisés dans les bases scientifiques comme EcoTaxa sont généralement entraînés sur plusieurs milliers d'images. Notre objectif était ici de tester la faisabilité de la méthode plutôt que d'obtenir un modèle parfaitement généralisable. Les images utilisées pour l'entraînement proviennent de bases de données scientifiques accessibles en ligne, en particulier EcoTaxa (base de données limitée pour les utilisateurs gratuits). Nous avons sélectionné celles présentant clairement les caractéristiques morphologiques des zoées.

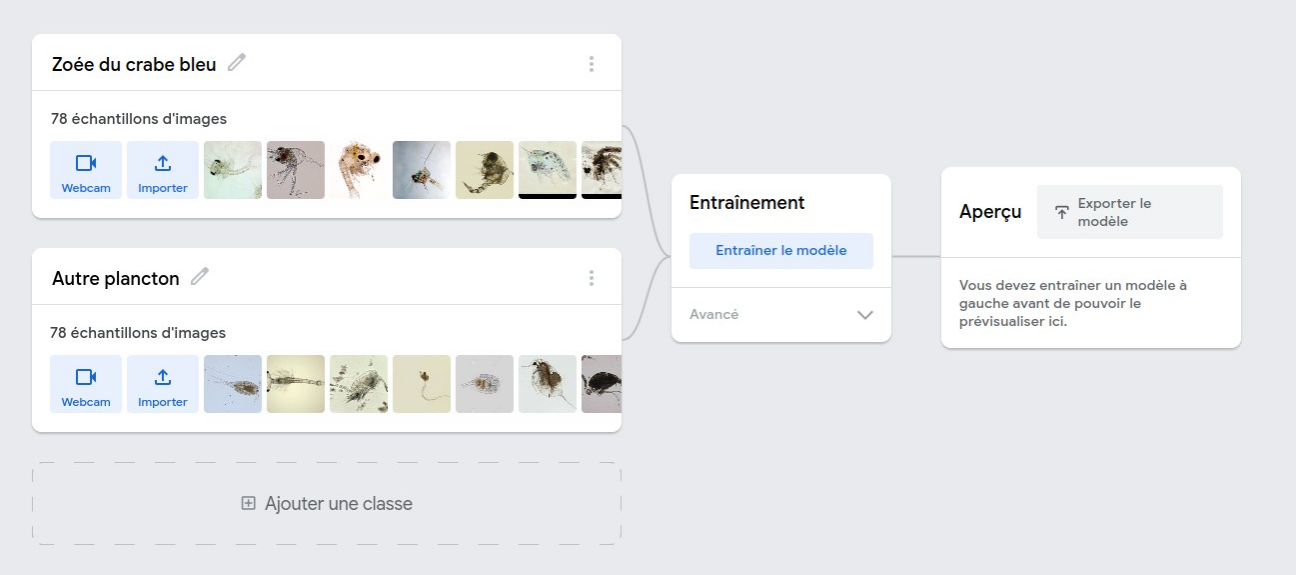


Fig. 9 — Création des classes et chargement des images dans Teachable Machine

3.3 Résultats de l'entraînement et tests

Une fois les images chargées, nous avons lancé l'entraînement. Teachable Machine affiche en temps réel la progression. Il est possible de changer certains paramètres, par exemple le nombre d'utilisations de chaque image et le temps passé sur chaque image. Nous avons ensuite testé le modèle avec de nouvelles images non utilisées pendant l'entraînement, pour voir si l'entraînement avait fonctionné.

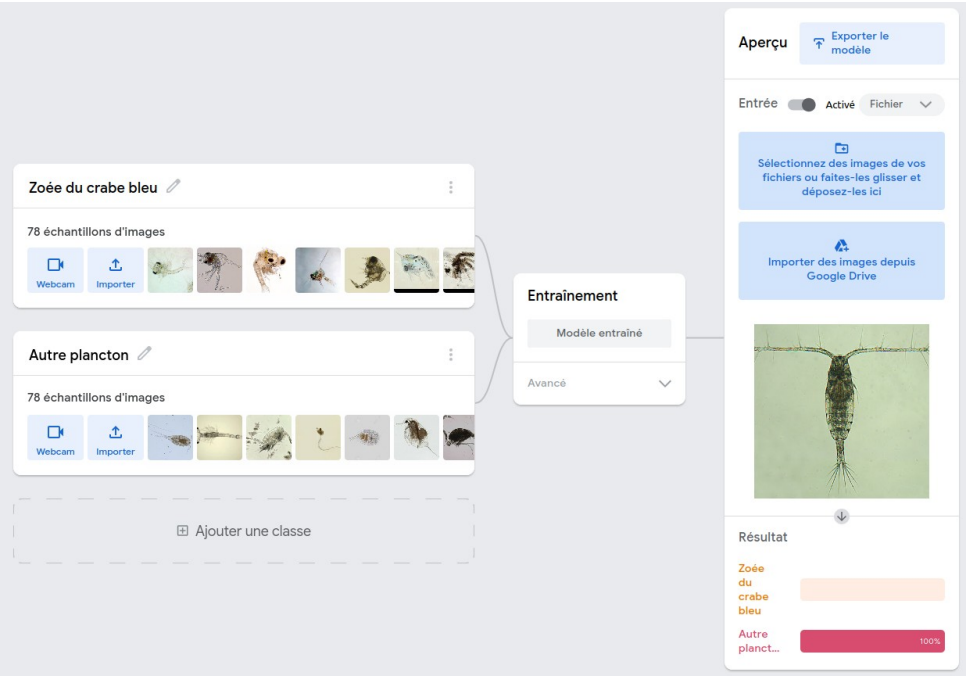


Fig. 10 — Résultats de l'entraînement et précision du modèle

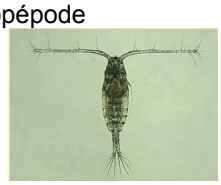
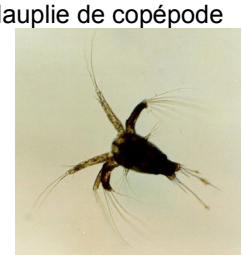
Image testée	Résultat IA	Correct ?	Analyse du résultat
Zoée (fond blanc, stade précoce) 	**Zoée — 94%**	Oui	Conditions idéales : fond clair, zoée bien reconnaissable.
Zoée (fond sombre) 	**Autre plancton — 68%**	Non	Le fond sombre n'est pas représenté dans le jeu de données d'entraînement. La probabilité de classification est faible, ce qui traduit une incertitude du modèle.
Copépode 	**Autre plancton — 100%**	Oui	L'IA reconnaît bien les organismes qui ne sont pas des zoées.
Stade larvaire avancé (mégalope) 	**Autre plancton — 75%**	Non	Ce stade larvaire est absent de l'entraînement et la morphologie est différente. Le pourcentage est moins élevé que précédemment, des éléments tels que les yeux ou le corps partagent des similarités avec la zoée.
Nauplie de copépode 	**Autre plancton — 98%**	Oui	Forme bien distincte de la zoée, correctement classée.

Tableau 2 — Résultats des tests après l'entraînement de l'IA

Ainsi, la base d'entraînement initiale pourrait peut-être être enrichie d'images avec des fonds différents, ainsi que des stades larvaires plus avancés si on veut détecter des zoées plus âgées. Mais cela va comporter des risques : si on ajoute des fonds sombres, il faut varier (pas noirs uniquement, mais gris plus ou moins foncés). Cela va aussi jouer sur la couleur du corps du plancton, qui sera blanc au lieu de gris, et l'IA pourrait avoir du mal à l'interpréter. Si on introduit des zoées plus développées, il faudra suffisamment d'exemples différents pour que l'IA trouve des points communs avec la zoée précoce, et ces organismes sont tout de même très différents. Créer une nouvelle classe « mégalope » serait plus judicieux. Comme nous travaillons uniquement sur fond clair, et avec des larves au stade zoée uniquement de taille maximale 300 µm (à cause de la taille du capillaire), nous n'avons pas modifié notre banque d'images.

3.4 Export du modèle et intégration Python

Une fois les performances jugées satisfaisantes, Teachable Machine permet d'exporter le modèle. Nous avons choisi le format TensorFlow / Keras (.h5), compatible avec Python. Ce fichier, que nous avons nommé plancton_model.h5, contient l'IA entraînée à reconnaître la zoée du crabe bleu. Il peut être chargé à tout moment dans un script Python avec la bibliothèque TensorFlow/Keras pour réaliser des prédictions sur de nouvelles images.

IV. Développement des scripts Python

Parallèlement au travail sur l'IA, nous avons développé une chaîne de traitement en Python pour automatiser l'extraction et l'analyse des images produites par le planktoscope. Cette chaîne comprend trois scripts, dont l'un (la génération d'une vidéo de test) n'est pas utilisé dans le traitement réel des échantillons, mais a joué un rôle important dans le développement et la mise au point des deux autres.

4.1 Script 1 : simuler le flux du planktoscope

Avant de disposer d'un vrai échantillon de plancton, nous avions besoin d'un support pour tester et mettre au point nos scripts de détection et d'analyse. Plutôt que d'attendre un prélèvement, nous avons écrit un premier script Python qui génère une vidéo de 2 minutes simulant, de façon simplifiée, ce que produit le planktoscope.

Ce script étant très complexe, nous avons eu recours à un service d'IA (ChatGPT) pour le rédiger. Nous avons présenté à l'IA ce que nous voulions comme vidéo et avons échanger jusqu'à ce que la vidéo générée soit satisfaisante. Ce script est donc un outil de développement, et non une partie de la chaîne de traitement des échantillons réels. Il nous a permis de rédiger et tester les scripts 2 et 3 dans de bonnes conditions, sans dépendre du prélèvement d'échantillons. Il n'est donc pas reproduit dans ce rapport.

La vidéo générée présente des formes géométriques (cercles, rectangles et triangles) qui défilent du haut vers le bas de l'écran sur un fond blanc, de la même manière que les organismes planctoniques s'écoulent dans le capillaire devant la caméra. Chaque forme est numérotée, de taille aléatoire, et se déplace avec une légère oscillation pour imiter la dérive des organismes dans le capillaire. La qualité de la vidéo est volontairement imparfaite : un flou aléatoire et des variations de luminosité sont ajoutés pour se rapprocher des conditions réelles d'un microscope.

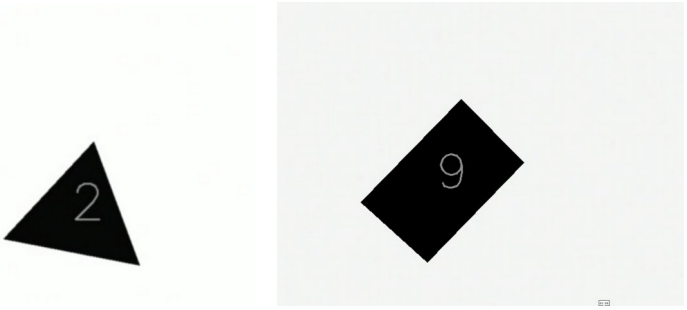


Fig. 11 — Formes géométriques numérotées défilant sur fond clair. La vidéo est visible en suivant le QRCode.

Pour valider notre chaîne de traitement sur la vidéo de test, nous avons également entraîné une version de démonstration de l'IA à reconnaître les formes géométriques (cercle, rectangle, triangle), exactement comme nous l'avons fait avec les zoées et le reste du plancton. Cette étape de validation sur des formes simples, avant de passer aux vrais organismes, nous a permis de nous assurer que la démarche était correcte.

4.2 Script 2 : détection et capture automatique

Ce script (detection_plancton.py commenté en annexe) est le cœur du traitement. Il analyse la vidéo image par image à l'aide de la bibliothèque OpenCV et extrait automatiquement chaque organisme (ou forme géométrique dans nos tests) qui passe devant la caméra.

Le traitement se fait en plusieurs étapes successives :

1	Conversion de l'image couleur en niveaux de gris, puis application d'un seuillage simple pour séparer les objets foncés (les organismes) du fond clair. Il a fallu travailler ce seuillage pour que tous les organismes soient détectés.
2	Détection des organismes : chaque groupe de pixels connectés est identifié comme un objet distinct. Les objets trop petits (moins de 500 pixels) ou trop proches des bords sont ignorés. Cela permet de ne détecter que les « vrais » organismes (et pas les débris) et d'avoir une image centrée exploitable.
3	Suivi temporel : chaque objet est suivi d'image en image grâce à sa position. Lorsqu'il franchit le centre de l'écran (ligne médiane), une capture est déclenchée et l'image est enregistrée dans un dossier sur l'ordinateur.
4	Extraction sur fond blanc : le contour précis de l'organisme est découpé et replacé sur un fond blanc uniforme, pour homogénéiser les images avant l'analyse par l'IA. Cette étape a nécessité plusiers allers-retours car les formes étaient parfois découpées de manière trop brutale, et il a fallu ajuster le seuil.

4.3 Script 3 : analyse par l'IA

Le troisième script (analyse_IA.py en annexe) charge le modèle entraîné (plancton_model.h5) et analyse en lot toutes les images extraites par le script 2 et placées dans le dossier. Pour chaque image, il calcule la probabilité qu'elle appartienne à la classe « zoée du crabe bleu ». Si cette probabilité dépasse un seuil de confiance fixé arbitrairement à 60 % (seuil par défaut de Teachable Machine), l'image est comptée comme une détection positive.

À la fin de l'exécution, le script affiche pour chaque image son score de confiance, ainsi que le nombre total de zoées détectées dans l'échantillon analysé.

4.4 Difficultés rencontrées et itérations

Le développement du script de détection (script 2) a nécessité plusieurs cycles d'itération avant d'aboutir à un résultat satisfaisant. Travailler d'abord sur la vidéo de test (avec ses formes géométriques simples et numérotées) nous a permis d'identifier et de corriger les problèmes progressivement.

Problème	Description détaillée	Solution apportée	Impact
Objets non détectés	Le seuil de gris initial (valeur fixe à 128) ne fonctionnait pas bien quand la vidéo présentait des variations de luminosité. Certaines formes passaient sans être détectées.	Ajustement du seuil de seuillage (passage à 200 sur fond clair)	Toutes les formes sont maintenant détectées de manière fiable.
Détections multiples	Lorsque la forme était en mouvement et présentait un léger flou, elle était parfois détectée plusieurs fois à la suite, produisant plusieurs captures du même objet.	Ajout d'un système de suivi par distance : un nouvel objet ne peut être créé que si aucun objet connu n'est déjà proche. Chaque objet ne peut être capturé qu'une fois.	Chaque forme n'est capturée qu'une seule fois, quelle que soit sa vitesse.

Contours mal extraits	Pour les formes avec des contours complexes ou irréguliers (notamment les rectangles légèrement déformés), le masque extrait était soit trop simplifié (perte de détails), soit bruité (pixels parasites inclus).	Ajout d'un nettoyage du masque avant extraction.	Images extraites plus propres et plus fidèles à la forme réelle.
Faux positifs (bruit)	Des variations aléatoires de luminosité étaient parfois interprétées comme de petits objets.	Filtrage par taille minimale (surface > 500 pixels) et par distance aux bords de l'image.	Élimination des faux positifs liés au bruit de l'image.
Plusieurs objets simultanés	Si deux formes se chevauchaient ou étaient très proches, elles étaient fusionnées en un seul objet.	Limitation du débit de la pompe (dans le cas réel) et paramètre de délai minimum entre deux apparations dans la vidéo de test.	Réduction significative des fusions d'objets.

Ces itérations successives sur la vidéo de test nous ont permis d'arriver à un script robuste avant de le confronter aux vrais échantillons microscopiques, qui présentent des organismes aux formes beaucoup plus irrégulières et variées que nos formes géométriques de test.

V. Premiers échantillons analysés

5.1 Collecte à l'Huveaune

Pour des raisons d'organisation et de météo, notre premier prélèvement de planctons a été effectué à l'Huveaune, une rivière qui traverse Marseille avant de se jeter en Méditerranée. Ce site est facilement accessible et nous a permis de tester l'ensemble de notre chaîne de traitement dans de vraies conditions.

Cette collecte préliminaire n'avait pas pour objectif de trouver des zoées de crabe bleu, car le crabe bleu se reproduit au printemps et ses larves ne sont présentes dans l'étang de Berre qu'à cette période (mai/juin). L'objectif était de voir comment réaliser un prélèvement puis de valider nos outils sur un échantillon biologique plutôt que sur notre vidéo de test.

> **À quand la sortie à l'étang de Berre ?**
> Le crabe bleu se reproduit au printemps. Ses larves ne sont présentes dans le plancton de l'étang qu'à cette saison, aux alentours de mai/juin. Il ne faut pas y aller trop tard car les larves seront alors à des stades trop avancés. Nous prévoyons deux sorties à l'étang de Berre avec l'association 8 Vies pour la planète : une première sortie mi-mai et une seconde début juin. Nous espérons pouvoir faire des prélèvements à différents endroits de l'étang pour voir si nous détectons des larves. Si nous en détectons, nous pourrons nous demander si elles ne vivent qu'à certains endroits de l'étang.

5.2 Filtrage et préparation des échantillons

La collecte de zooplancton suit un protocole pour obtenir un échantillon utilisable dans notre planktoscope.

- Collecte : un filet à plancton doit être utilisé pour prélever un échantillon. C'est un filet en forme de cône avec un robinet au bout. Ses mailles mesurent 100 µm. Elles peuvent être plus fines, mais comme nous voulons observer des larves de tailles supérieures, c'est suffisant. Le filet permet de concentrer les organismes dans l'échantillon. Il est immergé et traîné dans l'eau, lentement, avec plusieurs allers-retours. Il retient tous les organismes

dont la taille dépasse 100 µm, et ces organismes sont piégés dans la tube inférieur. Il suffit ensuite d'ouvrir le robinet pour collecter l'échantillon concentré.

Fig. 12 — Le filet à planctons, avec le robinet de prélèvement (jaune). En arrière plan, le spot de prélèvement.

- Filtrage préliminaire : nous avons fabriqué un petit entonnoir afin de filtrer l'échantillon, imprimé en 3D avec une imprimante PLA. Cet entonnoir est en deux parties qui se vissent et permettent de placer un filtre entre les deux. Il s'insère dans nos tubes de collecte et permet de filtrer l'échantillon. Nous avons choisi un filtre avec des mailles de 250 µm. Cela permet de retenir les débris et organismes trop grands qui pourraient boucher le capillaire de 300 µm.

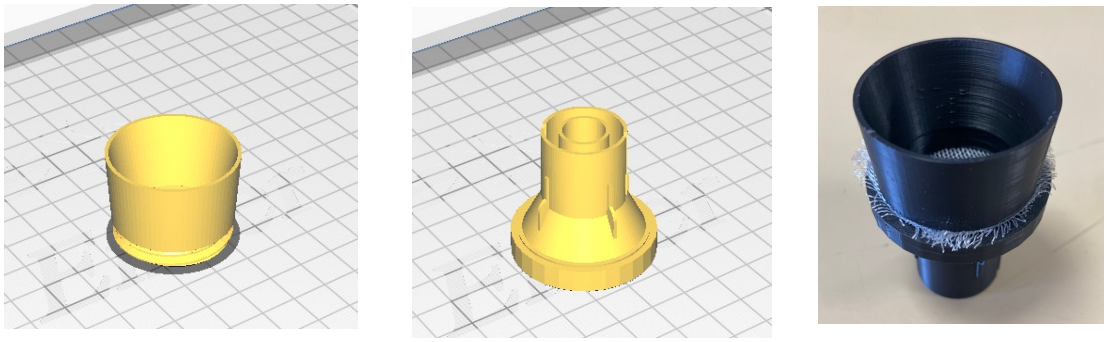


Fig. 13 — Modèles 3D et entonnoir imprimé

- Chargement dans le planktoscope : l'échantillon filtré (dont la taille des organismes est comprise entre 100 et 250 µm) est versé dans la seringue, et passe ensuite dans le circuit microfluidique du planktoscope.

Ce double filtrage garantit que les organismes observés ont une taille compatible avec le capillaire du planktoscope. La zoée du crabe bleu mesure entre 150 et 250 µm selon son stade, ce qui la place dans notre fenêtre de collecte.

5.3 Premiers résultats au microscope

Lors de notre premier passage d'échantillon dans le planktoscope, nous avons obtenu plusieurs images d'organismes planctoniques (phytoplancton et zooplancton) de l'Huveaune. La collecte s'est faite en bord d'eau, ce qui est moins efficace que depuis une embarcation, et à faible profondeur, ce qui explique qu'il n'y a pas beaucoup d'organismes. Les résultats sont tout de même très encourageants : le microscope fonctionne correctement, le flux microfluidique est stable et les images sont nettes. La Fig. 14 montre les captures d'écran lors du passage de 4 organismes dans le capillaire.

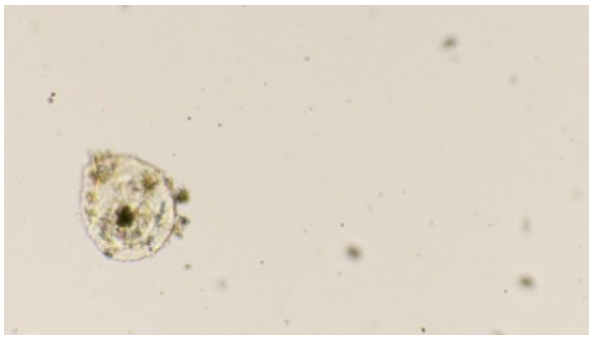


Fig. 14a — Organisme rond, probablement un protiste (~200 µm)

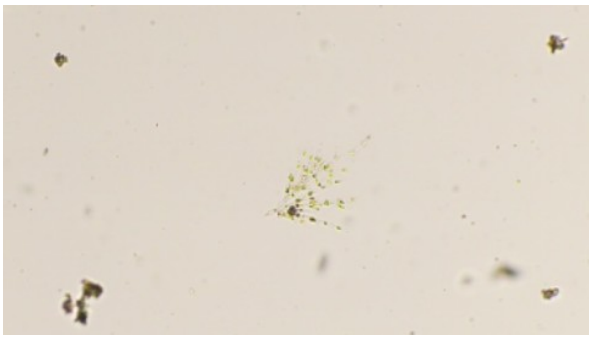


Fig. 14b — Organisme étoilé, probablement un amas de microalgues (~150 µm)

Fig. 14c — Organisme filamenteux, probablement un nématode (~300 µm)

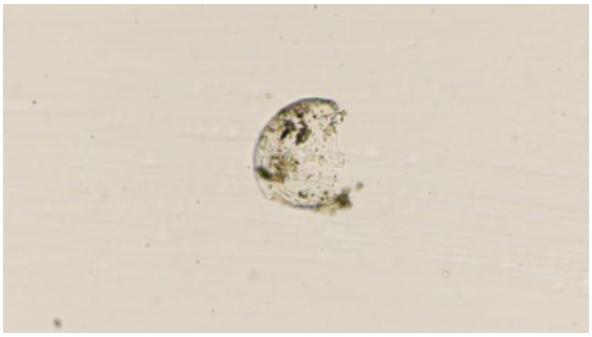


Fig. 14d — Organisme rond abîmé, probablement un protiste (~200 µm)

Le flux vidéo a ensuite été traité avec notre script Python afin d'extraire automatiquement les images de ces organismes et de « nettoyer » les images avant de pouvoir les analyser avec notre modèle IA. Les organismes sont pour la plupart correctement détectés et suivis. L'organisme de la Fig. 14b, un amas de microalgues, n'est pas détecté : il est probablement trop clair et en dessous du seuil. Le 14c non plus, probablement à cause de sa dimension latérale trop petite. Les deux autres organismes donnés en exemple (14a et 14d) sont détectés. Le « nettoyage » de l'image est pour le moment soit trop agressif, comme le montre la Fig. 15a, avec des contours correctement reproduits mais des détails fins qui disparaissent, soit trop permissif, comme le montre la figure 15b, ce qui pourrait induire l'IA en erreur. Nous travaillons à une solution pour l'améliorer.

Fig. 15 — Organisme de la Fig. 14a après extraction vidéo et nettoyage de l'image. L'extrait vidéo correspondant est visible en suivant le QRCode.

Dans l'attente de l'optimisation du script de nettoyage sur images réelles, nous avons soumis les images originales à l'IA à titre de test préliminaire.

Figure	Description	Identification probable	Résultat IA
Fig. 14a	Organisme rond, translucide, avec structure interne visible. Taille ~200 µm.	*Protiste*	Non 100 %
Fig. 14b	Petits appendices rayonnants de teinte verdâtre. Taille ~150 µm.	*Amas de microalgues*	Non 100 %
Fig. 14c	Forme filamenteuse allongée. Taille ~300 µm.	*Nématode*	Non 100 %
Fig. 14d	Forme courbe en C, carapace visible. Taille ~200 µm.	*Protiste dont la paroi a été abîmée*	Non 97 %

Ces résultats valident notre approche. Le planktoscope produit des images de qualité suffisante pour distinguer les organismes, et notre script de détection est parvenu à extraire automatiquement plusieurs d'entre eux. Pour le moment, le traitement d'image reste trop agressif, et l'image n'est pas exploitable. Cependant, le script permettant d'utiliser notre modèle d'IA fonctionne bien.

VI. Conclusion et perspectives

Bilan provisoire du projet

Ce projet nous a permis de mener une démarche complète, allant de la fabrication d'un instrument scientifique à l'analyse automatique d'images biologiques par intelligence artificielle. Voici les principaux résultats obtenus :

- Nous avons construit de A à Z le planktoscope Fairscope v2.6, en assemblant et soudant plus d'une centaine de pièces et en configurant l'ensemble électronique et logiciel. Le microscope fonctionne : les images obtenues lors de notre prélèvement à l'Huveaune sont nettes et permettent d'observer clairement les organismes planctoniques.

- Notre première sortie à l'Huveaune nous a permis de tester le protocole de prélèvement d'échantillons, en vue de notre première sortie à l'étang de Berre, ainsi que le matériel acheté (filet à planctons) ou conçu et fabriqué (entonnoir filtrant) pour l'occasion.

- Sur le plan de l'intelligence artificielle, nous avons entraîné un modèle capable de distinguer la zoée du crabe bleu des autres organismes planctoniques, avec un taux de reconnaissance supérieur à 90 % dans les conditions idéales. Ce score correspond au taux de classification correcte sur un petit ensemble d'images de test (moins de 20 images), ce qui reste indicatif. Les erreurs identifiées (fond trop sombre, stades larvaires trop avancés...) nous permettent déjà d'avoir des pistes d'amélioration de notre modèle.

- Côté programmation, notre chaîne de traitement Python (mise au point sur une vidéo synthétique avant d'être testée sur de vraies images) fonctionne en partie. Sur la vidéo «test», générée elle aussi grâce à un script Python, la chaîne de traitement est entièrement fonctionnelle. Sur nos premiers échantillons biologiques, les organismes sont correctement détectés pour la plupart. Les organismes non détectés possèdent des caractéristiques très éloignées de celles de la larve du crabe bleu et cela ne devrait donc pas empêcher sa détection. Le traitement de l'image extraite est pour le moment trop agressif et a du mal à rendre les détails les plus fins. Diminuer le seuil de traitement amène une confusion avec le fond de l'image et le résultat n'est pas satisfaisant. Enfin, le script permettant de traiter un ensemble d'images de manière automatique par notre IA

fonctionne parfaitement, et les organismes sont correctement classés. Nous pouvons donc vérifier qu'il n'y a pas de faux positifs, il faudra attendre de détecter des larves du crabe bleu au printemps pour s'assurer de la fiabilité réelle de notre modèle IA.

Réponse à la problématique

Notre problématique était la suivante : *comment détecter automatiquement la présence de larves de crabe bleu dans l'étang de Berre à l'aide de la microscopie et de l'intelligence artificielle ?*

Nous y avons apporté une réponse concrète en concevant un système fonctionnel en conditions de test. En combinant un microscope à flux continu (le planktoscope), un algorithme de détection d'objets basé sur OpenCV et un modèle d'intelligence artificielle entraîné avec Teachable Machine, il devient possible d'analyser automatiquement un échantillon de plancton et d'y repérer les zoées de crabe bleu, sans qu'un expert ait besoin d'examiner chaque image manuellement.

Notre projet se distingue par l'association d'un dispositif open-source et relativement accessible d'observation du plancton avec un outil d'intelligence artificielle capable d'identifier automatiquement les larves de crabe bleu. Cette approche illustre comment des technologies modernes peuvent être mobilisées pour améliorer la surveillance des espèces invasives et contribuer à une meilleure compréhension des écosystèmes aquatiques.

Perspectives et suite du projet

La prochaine grande étape est la sortie à l'étang de Berre, prévue au printemps 2026, lorsque le crabe bleu entre en période de reproduction. Ce sera la véritable mise à l'épreuve de notre système sur le site d'étude visé dans le cadre de ce projet.

À court terme :

- Prélèvement de plancton à l'étang de Berre au printemps, en collaboration avec l'association 8 Vies pour la planète et test complet de la chaîne de traitement sur un échantillon contenant potentiellement des zoées de crabe bleu (mai/juin)
- Amélioration du script 2 permettant le traitement d'images ;
- Enrichissement du jeu de données IA avec les nouvelles images obtenues.

À moyen terme :

- Réalisation de prélèvements répétés en différents points de l'étang pour cartographier les zones de ponte ;
- Corrélation entre la densité de larves détectée et les paramètres environnementaux (température, salinité, saison) ;
- Amélioration du script de détection pour mieux gérer les cas complexes (organismes superposés, formes très irrégulières).

À long terme :

- Mise à disposition de notre outil aux scientifiques et acteurs locaux pour un suivi régulier de la population de crabe bleu ;
- Contribution aux efforts de régulation de l'espèce invasive dans l'étang de Berre.

Ce que ce projet nous a apporté

Au-delà des résultats scientifiques, ce projet nous a appris à construire un instrument de nos mains, à comprendre concrètement le fonctionnement de l'intelligence artificielle, à programmer en Python pour résoudre un problème réel, et à travailler en équipe sur un projet réunissant biologie, physique, informatique et électronique.

Annexes — Scripts Python

Les deux scripts Python développés dans le cadre de ce projet et utilisé dans la chaîne de traitement sont reproduits ci-dessous dans leur intégralité, avec des commentaires par blocs expliquant les principales étapes. Le script de génération de la vidéo n'est pas reproduit car il ne sert pas en conditions réelles.

Script 2 — Détection et capture automatique (detection_plancton.py)

```
import cv2               # importation des bibliothèques
import numpy as np
import os
import math

############################################################
# BLOC 1 : Paramètres généraux du programme
# Ici on définit les réglages principaux : la vidéo à analyser,
# l'endroit où enregistrer les images extraites, et quelques
# seuils qui permettent de reconnaître et suivre correctement
# les objets dans la vidéo.
############################################################
video_path = "test_plancton.mp4"      # emplacement de la vidéo
output_dir = "captures"               # dossier où seront stockées les images
os.makedirs(output_dir, exist_ok=True)

min_object_area = 500          # taille minimale pour qu'un objet soit jugé intéressant
max_tracking_distance = 60     # distance max pour considérer qu'il s'agit du même objet
border_margin = 10             # zone à ignorer près des bords, moins fiable

############################################################
# BLOC 2 : Initialisation
# On ouvre la vidéo et on prépare les variables qui permettront
# de suivre plusieurs objets au fil des images (mémoire du suivi,
# numéro d'identifiant, compteur de captures).
############################################################
cap = cv2.VideoCapture(video_path)
tracked_objects = []
next_id = 1
capture_count = 0

############################################################
# BLOC 3 : Lecture de la vidéo image par image
# On traite chaque image successivement, jusqu'à la fin de la vidéo.
############################################################
while True:
    ret, frame = cap.read()
    if not ret:   # fin de la vidéo
        break

    H, W = frame.shape[:2]
    center_y = H // 2   # ligne horizontale qui sert de déclencheur

    ##########################################################
    # BLOC 4 : Prétraitement léger pour faire ressortir les objets
    # On convertit l'image en noir et blanc, puis on détecte
    # les zones sombres sur fond clair. Ensuite on applique un
    # petit nettoyage pour enlever les points parasites.
    ##########################################################
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    _, thresh = cv2.threshold(gray, 200, 255, cv2.THRESH_BINARY_INV)

    kernel = np.ones((3,3), np.uint8)
    thresh = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=1)
    thresh = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, kernel, iterations=1)
```

```
    ############################################################
    # BLOC 5 : Détection des objets dans l'image
    # On repère les différentes zones blanches (objets) et on
    # mesure leur taille, leur contour et leur emplacement.
    # On ne garde que les objets suffisamment grands et situés
    # dans une zone fiable de l'image.
    ############################################################
    num_labels, labels, stats, centroids = cv2.connectedComponentsWithStats(
        thresh, connectivity=8
    )

    detections = []

    for i in range(1, num_labels):  # on ignore l'arrière-plan
        area = stats[i, cv2.CC_STAT_AREA]
        if area < min_object_area:
            continue

        cx, cy = centroids[i]
        cx, cy = int(cx), int(cy)

        mask = (labels == i).astype(np.uint8) * 255
        contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
        if not contours:
            continue

        cnt = max(contours, key=cv2.contourArea)

        # exclusion des objets proches des bords
        x, y, w, h = cv2.boundingRect(cnt)
        if (x <= border_margin or y <= border_margin or
            x + w >= W - border_margin or
            y + h >= H - border_margin):
            continue

        detections.append({
            'center': (cx, cy),
            'cnt': cnt
        })

    ############################################################
    # BLOC 6 : Correspondance entre objets détectés et objets suivis
    # L'objectif est d'associer chaque nouvel objet détecté avec
    # un objet déjà suivi, pour savoir si c'est le même ou non.
    # Si ce n'est pas le même, on crée un nouvel objet suivi.
    ############################################################
    updated = []

    for det in detections:
        cx, cy = det['center']
        matched = False

        for obj in tracked_objects:
            ox, oy = obj['center']
            if math.hypot(cx - ox, cy - oy) < max_tracking_distance:
                obj['center'] = (cx, cy)
                obj['cnt'] = det['cnt']
                matched = True
                updated.append(obj)
                break

        if not matched:
            updated.append({
                'id': next_id,
                'center': (cx, cy),
                'last_y': cy,
                'cnt': det['cnt'],
                'captured': False
            })
            next_id += 1

    tracked_objects = updated
```

```
    ############################################################
    # BLOC 7 : Déclenchement des captures
    # Un objet est capturé au moment où il traverse la ligne
    # centrale de l'image. On isole alors sa forme et on
    # l'enregistre sur fond blanc.
    ############################################################
    for obj in tracked_objects:
        cx, cy = obj['center']

        # passage du dessus vers le dessous
        if (not obj['captured'] and obj['last_y'] < center_y <= cy):

            label = labels[cy, cx]
            if label == 0:
                continue

            component_mask = (labels == label).astype(np.uint8) * 255
            component_mask = cv2.morphologyEx(component_mask, cv2.MORPH_CLOSE, kernel,
iterations=1)

            obj_only = cv2.bitwise_and(frame, frame, mask=component_mask)
            bg_white = np.ones_like(frame) * 255
            isolated = np.where(component_mask[...,None] == 255, obj_only, bg_white)

            cv2.imwrite(os.path.join(output_dir, f"capture_{capture_count:03d}.png"),
isolated)

            capture_count += 1
            obj['captured'] = True

        obj['last_y'] = cy

    ############################################################
    # BLOC 8 : Nettoyage des objets sortis du champ
    # Pour éviter d'accumuler trop d'objets en mémoire, on
    # supprime ceux qui sont définitivement sortis de l'image.
    ############################################################
    tracked_objects = [
        o for o in tracked_objects
        if o['center'][1] < H + 50
    ]

################################################################
# BLOC 9 : Fin du programme
# Fermeture de la vidéo et affichage du nombre total
# d'objets capturés.
################################################################
cap.release()
cv2.destroyAllWindows()
print(f"Traitement terminé. {capture_count} planctons détectés.")
```

Script 3 — Analyse par l'IA (analyse_IA.py)

```
import os                               # importation des bibliothèques
import numpy as np
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image

############################################################
# BLOC 1 : Paramètres généraux
# Définition du modèle IA, du dossier contenant les images
# à analyser, de la taille standard des images, et du seuil
# de confiance nécessaire pour valider une détection.
############################################################
model_path = "plancton_model.h5"        # importation du modèle Teachable Machine
images_dir = "captures"                 # dossier où sont les images à analyser
```

```
IMG_SIZE = (224, 224)
target_class_index = 0
confidence_threshold = 0.6

################################################################
# BLOC 2 : Chargement du modèle IA
# Le modèle a été entraîné au préalable pour reconnaître
# différents types d'objets, ici la larve du crabe bleu
################################################################
model = load_model(model_path, compile=False)

################################################################
# BLOC 3 : Récupération des images à analyser
# On liste toutes les images présentes dans le dossier de
# captures qui correspondent aux formats courants.
################################################################
image_files = [
    f for f in os.listdir(images_dir)
    if f.lower().endswith(('.png','.jpg','.jpeg'))
]

matched_images = []

print("Analyse des images...")

################################################################
# BLOC 4 : Analyse IA de chaque image
# Chaque image est chargée, redimensionnée puis
# envoyée au modèle IA pour obtenir une prédiction et une
# probabilité associée.
################################################################
for img_file in image_files:
    img_path = os.path.join(images_dir, img_file)
    img = image.load_img(img_path, target_size=IMG_SIZE)
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0) / 255.0

    preds = model.predict(img_array, verbose=0)[0]
    class_idx = np.argmax(preds)
    confidence = preds[target_class_index]

    print(f"{img_file} → {preds} (classe={class_idx}, confiance={confidence:.3f})")

    ############################################################
    # BLOC 5 : Filtrage des détections pertinentes
    # Si l'image est classée comme appartenant à la classe
    # recherchée, et avec une confiance suffisante, on la
    # conserve pour les résultats finaux.
    ############################################################
    if class_idx == target_class_index and confidence >= confidence_threshold:
        matched_images.append((img_file, confidence))

################################################################
# BLOC 6 : Résultat final
# On affiche le nombre total d'images reconnues et leur
# probabilité associée.
################################################################
print(f"\nNombre détecté : {len(matched_images)}")
for f, conf in matched_images:
    print(f"{f} - {conf*100:.1f}%")
```