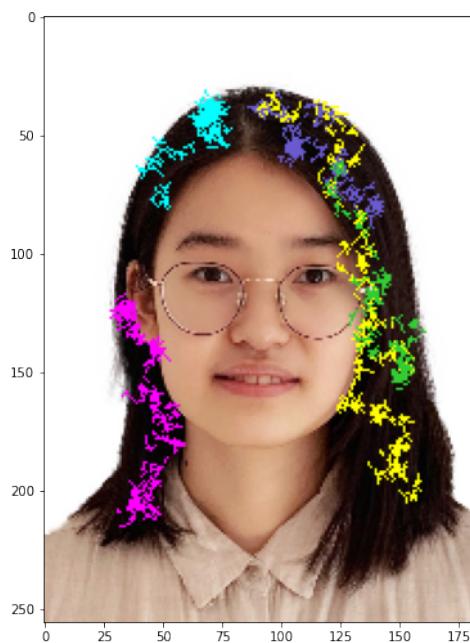


# Intelligence en essaim sur l'application de la segmentation des images

*Projet de résolution de problèmes en IA*

---



WANG Huaijin, XIONG Chen, YAN Tonglin  
GM5

6 mars 2022

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Méthodologie</b>	<b>3</b>
2.1	Intelligence en essaim . . . . .	3
2.1.1	La méthode proposée . . . . .	3
2.2	Algorithme araignée sociale . . . . .	7
2.2.1	Coordination et structuration d'activités par le biais de l'environnement : une approche biologique . . . . .	7
2.2.2	Modélisation informatique . . . . .	7
2.2.3	Mise en oeuvre pour la construction d'une toile d'araignée virtuelle	8
2.2.4	Description du modèle . . . . .	9
2.2.5	Détail des comportements . . . . .	9
<b>3</b>	<b>Application</b>	<b>12</b>
<b>4</b>	<b>Implémentation</b>	<b>13</b>
4.1	Modélisation . . . . .	13
4.1.1	Intelligence en essaim . . . . .	13
4.1.2	Algorithme araignée sociale . . . . .	15
<b>5</b>	<b>Résultat</b>	<b>18</b>
5.1	Intelligence en essaim . . . . .	18
5.1.1	Nombre d'itérations . . . . .	19
5.1.2	La valeur de seuil $seuil_f$ . . . . .	20
5.2	Algorithme des araignées sociales . . . . .	20
5.2.1	Nombre d'itérations . . . . .	22
5.2.2	Probabilité de retour au toile . . . . .	22
5.2.3	Sélectivité . . . . .	23
<b>6</b>	<b>Conclusion</b>	<b>25</b>

# Chapitre 1

## Introduction

Depuis quelques années, les chercheurs conçoivent de nouveaux types de calculs avec des techniques inspirées de la biologie. Il s'agit de puiser dans les comportements des êtres vivants de nouvelles approches pour la résolution des problèmes courants. Avec l'intérêt sur ce domaine, nous allons faire une étude sur les approches d'intelligence en essaim, surtout l'algorithme de l'araignée sociale, et les appliquer dans la segmentation des images.

Nous allons commencer par la présentation des approches dans la vie artificielle, y compris le principe et l'algorithme en détail. Ensuite, nous allons expliquer comment les appliquer dans la segmentation des images. Avant de l'implémentation, nous introduisons la modélisation globale sur ce projet, et terminons par une démonstration du résultat.

# Chapitre 2

## Méthodologie

### 2.1 Intelligence en essaim

L'intelligence en essaim provient de l'étude du comportement en essaim des insectes sociaux représentés par les fourmis et les abeilles. Il a d'abord été utilisé dans la description des systèmes robotiques cellulaires. Son contrôle est distribué et il n'y a pas de contrôle central car le groupe d'insecte s'auto-organise.

Le principe de base de l'approche de conception de l'intelligence en essaim est d'adapter les mécanismes de coordination observés dans les systèmes biologiques distribués en systèmes d'ingénierie. Stigmergie est le plus connu et le plus puissant de ces mécanismes de coordination, son principe fondamental est que l'agent ne conçoit pas le travail pour lui-même, il est uniquement guidé par le travail.

Suivant le principe incrémental, cette méthode [1] nous permet de classifier les cellules en notion de Multi-agent.

Nous construisons d'abord un modèle basé sur la croissance du modèle incrémental régional. Ce modèle a toujours un ensemble de cellules potentielles qui peuvent être regroupés dans la zone afin qu'ils puissent être évalués par rapport à l'orientation de la vie artificielle et à la meilleure cellule sélectionnée.

#### 2.1.1 La méthode proposée

On place d'abord aléatoirement un certain nombre d'agents sur les cellules, puis ces agents peuvent se déplacer aléatoirement vers leurs huit cellules adjacentes et marquer ces cellules.

Le marquage des cellules par l'agent suit les principes suivants :

Si la valeur de l'ancienne position (cellule) de l'agent est proche de la valeur de la cellule à la position actuelle de l'agent (c'est-à-dire que la différence entre ces deux valeurs est

inférieure à un seuil), l'ancienne position (cellule) de l'agent garde sa marque , sinon il perd son marquage.

Après cette phase de mouvement aléatoire, on peut l'appeler phase d'incoordination (car le mouvement de l'agent est une incoordination). Certaines régions marquées apparaissent dans l'environnement, et ces régions deviennent comme des attracteurs, attirant l'agent vers les cellules qui les entourent.

Chaque zone génère un champ de phéromones qui se propage dans l'environnement local correspondant aux cellules entourant la zone marquée.

Ainsi, dans l'environnement, il y a autant de zones où la phéromone se propage que de zones marquées. Les phéromones différencient une zone d'une autre par leur odeur. La valeur de la concentration de phéromones dans la cellule proche de la zone marquée est une mesure qui indique si la cellule est adéquate par rapport à la zone marquée.

L'adéquation d'une cellule à une zone marquée est mesurée par deux critères : le critère d'homogénéité et le critère de compacité.

**Définition 1. Les mesures d'adéquation** Soient  $R(p) = (p_1, p_2, \dots, p_m)$  la région marquée (région d'attracteur) ;  $V(p) = (p_1, p_2, \dots, p_n)$  ; la région de cellules voisines.(4-connexe) de la région  $R(p)$  ;  $I(x, y)$  : l'intensité de  $p(x, y)$ . La valeur moyen de la région marquée  $R(p)$  est calculé :

$$\bar{I}_{R(p)} = \frac{I}{m} \sum_{i=1}^m I_{R(p)}(x_i, y_i) \quad (2.1)$$

L'évaluation de l'homogénéité d'une cellule  $p(x, y)$  de l'intensité  $I(x, y)$  de la région  $V(p)$  par rapport à la région  $R(p)$ , fournira une valeur  $E_1(p)$

$$E_{1(p)} = 1 - \min(1, \frac{|I(x, y) - \bar{I}_R|}{k}) \quad (2.2)$$

$k$  : une constante permettant de normaliser l'expression dans l'intervalle  $[0, 1]$ .

Les relations de voisinage (8-connexité) comme :

$$N_8(p(x_1, y_1), p(x_2, y_2)) = 1 \text{ si et seulement si } \max(|x_2 - x_1|, |y_2 - y_1|) = 1 \quad (2.3)$$

L'évaluation de compacité d'une cellule  $p(x, y)$  de la région  $V(p)$  par rapport à la région  $R(p)$  est donnée par :

$$E_2(p) = \frac{1}{8} \sum_{i=1}^8 N_8(p(x_i, y_i), p(x, y)) \quad (2.4)$$

L'évaluation globale d'une cellule  $p(x, y)$  de la région  $V(p)$  par rapport à la région  $R(p)$  est donnée par :

$$E_g(p) = \alpha E_1(p) + \beta E_2(p), \alpha + \beta = 1, \alpha; \beta \in [0, 1] \quad (2.5)$$

La région de propagation des phéromones est l'ensemble des cellules situées au voisinage (connectivité 4) de la région marquée (région attractrice), dont la mesure d'adéquation est supérieure au  $seuil_f$ .

**Définition 2. Région de propagation de phéromone** Soient  $R(p) = (p_1, p_2, \dots, p_m)$  la région marquée contenant  $m$  cellules :  $p_1, p_2, \dots, p_m$ ;  $V(p) = (p_1, p_2, \dots, p_n)$ ; la région de cellules voisines.(4-connexe) de la région  $R(p)$ ;  $E_g(p_i)$  : mesure d'adéquation de la cellule  $p_i$  à la région  $R(p)$ ;  $seuil_g$  : seuil d'adéquation d'une cellule de  $V(p)$  à la région  $R(p)$ .

La région de phéromone  $C(p) = (p_1, p_2, \dots, p_n)$  est définie :

$$C(p) \subseteq V(p) \text{ et } \forall p_1 \in C(p) \quad E_g(p_1) \geq seuil_f \quad (2.6)$$

Nous laissons le rapport entre la mesure d'adéquation de la cellule à la zone marquée Eg et la somme du nombre de cellules dans la zone de propagation de la phéromone et la zone marquée (la zone qui a produit la phéromone) définir comme la concentration de phéromone de chaque point (cellule) de la zone de propagation.

**Définition 3. Concentration de phéromone** Soient  $R(p) = (p_1, p_2, \dots, p_m)$  la région marquée contenant  $m$  cellules :  $p_1, p_2, \dots, p_m$ ;  $V(p) = (p_1, p_2, \dots, p_n)$ ; la région de cellules voisines.(4-connexe) de la région  $R(p)$ ;  $E_g(p_i)$  : mesure d'adéquation de la cellule  $p_i$  à la région  $R(p)$ .

La concentration de la phéromone dans une cellule  $p_i$  de la région  $C(p)$  est  $s(p_i)$ , telle que :

$$s(p_i) = \frac{E_g(p_i)}{m + n} \quad (2.7)$$

La présence de zones marquées (images) dans l'environnement a un effet sur le comportement de l'agent.

Un agent dont le mouvement est aléatoire dirigera son mouvement vers l'une des régions marquées que d'autres agents désignent comme modèles latents.

En même temps, l'agent sera repoussé par la grande densité d'agents.

L'agent mobile choisit probabiliste une des cases adjacentes à sa position (8 connectées), qui est couverte par le champ de phéromones.

La sélection des probabilités se fait en deux étapes.

Premièrement, l'agent juge la concentration en phéromones de chaque région adjacente, la concentration en phéromones d'une région est égale à 0, si la région adjacente n'appartient à aucune région de propagation de phéromones. Si la zone appartient à plusieurs zones de propagation de phéromones, et qu'elle a plusieurs valeurs de concentration de phéromones, on prend la concentration maximale.

Dans la deuxième étape, l'agent détermine l'attractivité relative d'une cellule car sa concentration locale en phéromones est normalisée par les concentrations de toutes les cellules voisines à son emplacement.

**Définition 4. Probabilité de sélection** Soient  $p(x_A, y_A)$  l'emplacement de l'agent A ;  $V_A(p)$  l'ensemble de voisinage (8-connexité) de la cellule  $p(x_A, y_A)$ . Selon (2.3),  $\forall p_i \in V_A(p)$ ,  $\max(|x_A - x_i|, |y_A - y_i|) = 1$  ;  $s_i$  : la concentration de phéromone dans la cellule  $p_i$

L'attraction relative de la cellule  $p_i$  de  $V_A(p)$  est :

$$f_i = \frac{s_i}{\sum_{p_j \in V_A(p)} (s_j)} \quad (2.8)$$

La probabilité de choisir une cellule  $p_i$  de  $V_A(p)$  est :

$$prob_i = \max_{p_j \in V_A(p)} \frac{f_i}{8} \quad (2.9)$$

Lorsque l'agent se déplace sur une case couverte par un champ de phéromones, la couleur de son marqueur s'adapte à la couleur de la zone qui a produit cette phéromone.

Le déplacement du substitut vers une région appartenant à plusieurs régions de propagation de phéromones permet de fusionner ces régions avec les régions marquées qui ont produit ces phéromones.

Un agent meurt si les cellules à côté de son emplacement actuel sont toutes marquées et non couvertes par des champs de phéromones. La mort d'un agent provoque l'initialisation d'un autre agent à un emplacement aléatoire dans la région de propagation des phéromones avec la plus faible densité d'agents.

Lorsque l'agent se déplace vers une cellule de la zone de propagation des phéromones (sélectionnée par probabilité), il la marque. Par conséquent, la cellule est retiré de la zone de propagation des phéromones et ajouté à la zone marquée.

Ainsi, la propagation de la phéromone sera étendue à un certain nombre de cellules voisines (4-connectivité) de la localisation actuelle de l'agent, et la valeur de sa mesure d'adéquation n'est supérieure qu'à un seuil  $seuil_f$ . Cette propagation se traduit également par une augmentation de la taille de la zone de propagation des phéromones. La diminution et l'augmentation de la taille de l'aire de diffusion des phéromones au cours du temps est interprétée comme une opération d'évaporation et de dispersion des phéromones.

On définit le facteur de diffusion des phéromones par le nombre de cellules ajoutées à la zone marquée, un facteur égal à zéro empêche toute diffusion. Et le facteur d'évaporation est réduit par le nombre de cellules dans la zone transmise par les phéromones.

## 2.2 Algorithme araignée sociale

### 2.2.1 Coordination et structuration d'activités par le biais de l'environnement : une approche biologique

La théorie de la stigmergie a été définie dans le cadre de l'activité bâtisseuse des termites et se résume ainsi : "la coordination des tâches, la régulation des constructions ne dépendent pas directement des ouvriers, mais des constructions elles-mêmes. L'ouvrier ne dirige pas son travail, il est guidé par lui"

La stigmergie peut être réalisée de deux manières très différentes, la stigmergie qualitative et la stigmergie quantitative. La stigmergie qualitative correspond à une succession de stimuli-réponses qualitativement différents. Cette forme de stigmergie est utilisée pour la simulation d'un processus de construction collective de nids de guêpes. Leur modèle se fonde sur un ensemble de règles comportementales déterministes (quelques dizaines de règles) associées à ce que les auteurs appellent à des configurations stimulantes. Dans la stigmergie quantitative, la répétition de stimuli-réponses diffèrent quantitativement. Il s'agit d'un mécanisme de feed-back positif ou amplificateur. (cf l'article[2])

### 2.2.2 Modélisation informatique

L'objectif de modélisation est pour construire la phénomène stigmergique. Le modèle peut être composé de la manière suivante :

1. Le comportement de l'agent est tel que celui-ci est supposé ne pas être spécialisé (par rapport à ses congénères) tous les agents peuvent donc effectuer individuellement la tâche impartie au système.
2. Le processus stigmergique correspond à une prise en compte dans ce comportement individuel des actions des autres agents (et de l'individu lui-même) par le biais des modifications sur l'environnement. La présence d'autres agents modifie quantitativement le comportement d'un autre agent, c'est à dire qu'il ne change pas de comportement en développant de nouvelles compétences mais qu'il change simplement les fréquences avec lesquelles il produit certaines actions.

**Comportement d'un agent** Le comportement d'un agent isolé est sous la forme de couples action-probabilité. Soit  $A = \{A_0, A_1, \dots, A_m\}$  l'ensemble des actions auxquelles un agent peut effectuer ;  $P : A \rightarrow [0, 1]$ , la probabilité d'effectuer une action donnée ; le couple  $(A_0, P(A_0))$  décrit la probabilité d'un agent qui effectuer l'action  $A_0$ .

### 2.2.3 Mise en oeuvre pour la construction d'une toile d'araignée virtuelle

On réalise l'expérimentation biologique sous la forme informatique. L'environnement est modélisé sous la forme d'une grille carrée, chaque case étant associée à un piquet de hauteur à valeur dans bas, moyen, haut.

L'agent(une araignée) ne peut faire que deux types actions indépendantes : se déplacer sur une base voisine et poser un fil de soie. Ces deux actions peuvent être fait en même temps par agent.

#### Comportement d'individuel simple

On suppose d'abord le comportement de déplacement dans l'une des cases adjacentes contient huit actions avec la même probabilité. Une action de probabilité constante dans le temps et dans l'espace : poser un fil. Dans le modèle on a donc  $A = \text{Nord, Sud, Est, Ouest, NordEst, NordOuest, SudEst, SudOuest}$  et  $P(a) = \frac{1}{8}, \forall a \in A$ . Le comportement lors de la simulation peut se ramener aux règles comportementales suivantes exécutées en séquence et indéfiniment :

1. Choisir une direction au hasard
2. Poser aléatoirement un fil

#### Comportement stigmergique

Nous introduisons maintenant dans notre modèle de comportement l'influence de l'environnement. Lorsqu'un fil de soie est posé il permet de rejoindre une autre case pas forcément adjacente. Le comportement est de l'araignée peut être de deux types : suivre un fil ou aller sur une case adjacente. On pose  $P_{fil}$  la probabilité d'emprunter d'avoir un comportement de type suivre un fil. On obtient alors un comportement qui est conditionné par le comportement antérieur de l'agent lui-même ou de ses congénères du fait de la présence des fils. La nouvelle règle de comportement de déplacement est :

- choisir aléatoirement le type de comportement
- dans le cas suivre un fil calculer la probabilité  $P_{filée}$  de suivre l'un des fils selon la formule  $P_{filée} = P_{fil}/\text{Nbfil}$  avec Nbfil non null le nombre de fils partant de la case courante et calculer les cases accessibles
- dans le cas aller sur une case adjacente ou Nbfil= 0 calculer la probabilité de se rendre sur chaque case adjacente  $\implies P_{adjacente} = (1 - P_{fil}) * \frac{1}{8}$

Attention : Si Nbfil = 0 ; on est ramené au cas de la règle simple de déplacement(voir règle1). (Référence [3])

### 2.2.4 Description du modèle

Les agents peuvent être répartis en ensembles. Dans ce cas, les agents appartenant au même ensemble sont se concentrant sur la même région et partageront les mêmes paramètres. Une distinction entre la soie fixée par les membres d'un groupe et la soie fixée par les membres d'autres groupes est fabriquée. Nous fournissons d'abord une vue globale du modèle d'agent qui est valable à la fois pour la région unique cas de détection et pour plusieurs cas de détection de régions. Les différences seront détaillées lorsque décrivant l'élément de mouvement.

**Environnement** L'environnement correspond à un tableau de dimension 2. Chaque case de tableau est caractéristique par la liste de fil (noté  $Dl_p$ ). Il n'y a aucun fil au départ. Chaque fil  $d$  dans  $Dl_p$  commence par une case  $p$  et arrive à une case finale  $f_d$ . Les draglines sont libellés par l'équipe d'araignée qui les crée, lorsqu'il y a plusieurs régions qui sont détecté simultanément. Les composants sont définies comme :

- Environnement= NM de case
- Case = [val : valeur de case ;D : Liste de fil]
- Dragline = [end,spiderg]

**Agents** Un agent a trois types de comportement :

1. Choisir un case adjacent et se déplacer
2. Choisir de fixer selon une probabilité contextuelle si la décision est prise, effectuer la fixation de la soie et quitter le cycle de base
3. Choisir à rentrer sur le web selon la probaDeRetour, si la décision est prise, retour à la dernière case fixée.

On définit trois fonctions qui nous aide de faire la perception de l'environnement et faire la décision des actions.

- $Neigh_p$  : respectivement offre la liste de cases au voisinage
- $Scuts_p$  : respectivement offre la liste de cases auxquels un agent peut arriver en suivant les draglines
- $Accessible_p$  : est la liste de cases qui contient l'union de deux premières

On définit aussi  $Number(a,b)$  avec  $a,b$  deux cases de l'environnement, comme le nombre de soies de  $a$  à  $b$ , et  $Number(a,b,spg)$  comme le nombre de soies libellé par l'équipe de agents. Ces deux valeurs va être utilisé pour choisir la comportement du déplacement dépendant l'attraction de soie.

### 2.2.5 Détail des comportements

Les comportements détaillent comment les agents interagissent avec leur environnement. Fondamentalement, nous calculons d'abord une probabilité de réaliser action à partir

des paramètres et des caractéristiques de la position courante, ou plus simple, on utilise une probabilité constante. Ensuite, selon cette probabilité, l'item comportemental est déclenché à décider l'action soit réalisé ou non. La partie suivante détaille le calcul de la probabilité et l'effet de l'action sur l'environnement et sur l'état interne.

**Déplacement** La soie est attractive et influence le mouvement des araignées : les araignées ont tendance à suivre les fils en soie plutôt que de se déplacer vers une position adjacente. On définit le poids de chaque case accessible selon la manière y accéder :

- Cases voisines ont le même poids
- Cases accessibles avec les draglines ont le poids selon le nombre de draglines les reliant à la position actuelle et à l'attraction de la soie

En normalisant les poids, on obtient une distribution de probabilités sur les positions accessibles. Il faut noter qu'une case peut être accessible simultanément par les deux sens. Dans un tel cas, son poids est la somme des deux poids décrits. Nous avons implémenté deux manières de calculer cette distribution de probabilités selon le nombre de régions à rechercher.

La première correspond à une situation dans laquelle des agents coopèrent pour détecter une seule région : Nous ne considérons qu'un seul ensemble d'agents et ne faisons aucune distinction dans la soie. Le choix du mouvement ne dépend pas du nombre de draglines observées et la décision peut être divisée en deux étapes. Tout d'abord, l'agent est confronté à une alternative : suivra-t-il ou non un fil en soie ?  $P_{dragline}$  est la probabilité pour l'agent de se déplacer dans cette direction. Si l'agent préfère suivre des draglines, il choisit aléatoirement un fil et atteint la case appartenant à  $Scuts_{CurrentP}$  à son extrémité. Ainsi, plus il y a de draglines menant à p, plus l'agent a de chances d'atteindre p. Sinon, il se déplace aléatoirement vers une case appartenant à  $Neigh_{CurrentP}$ .

1. Si  $p \in Neigh_{CurrentP}$ ,

$$Proba(\text{déplacement}(p)) = \frac{1 - P_{fil}}{\text{CardNb}(Neigh_{CurrentP})} \quad (2.10)$$

2. Si  $p \in Scuts_{CurrentP}$ ,

$$Proba(\text{déplacement}(p)) = \frac{P_{fil} * \text{Num}(CurrentP, p)}{\text{CardNb}(Dl_{CurrentP})} \quad (2.11)$$

La deuxième implémentation se place dans une perspective de compétition entre agents pour détecter plusieurs régions. Dans ce cas, on distingue la soie selon l'ensemble des agents qui l'ont tissée.

Évaluant la probabilité d'atteindre une case consiste à donner un poids constant à chaque case assignable dans  $Neigh_{CurrentP}$  et un poids lié au nombre de draglines entre cette case et la localisation de l'agent. Le choix de suivre un fil est alors dépendant du nombre de draglines présentes sur la case considérée et du type de soie. On distingue deux sortes

d'attractions pour la soie selon les labels des draglines. *AttractSelf* décrit l'attrirance d'un agent pour sa propre soie et pour la soie des membres de son groupe ; *AttractOther* décrit l'attrirance pour la soie des agents des autres groupes. À partir de là, on calcule deux contributions au poids d'une case qui se combinent.

$$Proba(\text{déplacement}(p)) = \frac{w(p)}{\sum w(a)} \quad (2.12)$$

où

- si  $p \in Neigh_{CurrentP}$ ,  
 $w(p) = \text{constante}$

(2.13)

- si  $p \in Scuts_{CurrentP}$ ,

$$\begin{aligned} w(p) = & AttractSelf \times F(\text{Number}(CurrentP, p, MyGroup)) \\ & + AttractOther \times F(\text{Number}(CurrentP, p) - \text{Number}(CurrentP, p, Mygroup)) \end{aligned} \quad (2.14)$$

$F$  est une fonction exprimant comment le nombre de draglines dans le chemin influence le poids jusqu'à une saturation donnée. Dans nos expériences, nous avons utilisé

$$F(x) = \min(x, SaturationValue) \quad (2.15)$$

*SaturationValue* correspond à une limite à partir de laquelle le nombre de draglines n'influence plus le résultat.

Une fois la décision prise, effectuant un mouvement consiste à mettre à jour la valeur courante de la case :  $CurrentP \leftarrow p$ .

**Poser un fil** La décision est prise en fonction de la probabilité de fixer un fil sur la case courante. Il est calculé à partir d'une distribution gaussienne dont la moyenne est *RefLevel* et dont l'écart type est  $1/Selectivity$ .

En fixant un fil consiste à ajouter un fil dans l'environnement ; cela se fait en mettant à jour la liste des draglines de la case courante  $Dl_{CurrentP} \leftarrow Dl_{CurrentP} \cup \{(CurrentP; LastFixed)\}$  et la liste des draglines de *LastFixed* case :  $Dl_{LastFixed} \leftarrow Dl_{LastFixed} \cup \{(LastFixed; CurrentP)\}$

**Retour au web** La probabilité de décision est constante (*BackProbability*) et la réalisation de l'action consiste à mettre à jour la localisation de l'agent :  $CurrentP \leftarrow LastFixed$

# Chapitre 3

## Application

Nous allons appliquer les approches présentées précédemment dans la segmentation des images, qui constitue le cœur du système de vision et une étape préliminaire dans toutes les tâches de vision.

La segmentation des images a pour l'objectif d'extraire les différents indices visuels tels que le contour des objets, les régions homogènes, et les objets 3D. Nous pouvons également la considérer comme une classification des cases.

Nous pouvons prendre des images au niveau gris ou des images colorées, en convertissant les valeurs RGB en valeur au niveau gris par la formulation

$$Grey = R * 0.299 + G * 0.587 + B * 0.114 \quad (3.1)$$

où R, G, B correspond à rouge, vert, bleu respectivement. L'environnement du système de multi-agent est constitué par la valeur au niveau gris en chaque pixel.

# Chapitre 4

## Implémentation

### 4.1 Modélisation

#### 4.1.1 Intelligence en essaim

**Class Abeille**

**Les attributs**

- *int x, int y* : La position dans l'environnement ;
- *Real intensité* : La valeur d'intensité apportée par cet abeille ;
- *Bool mort* : Cet abeille est mort ou pas ;
- *Int numr* : Cet abeille appartient à quelle région marquée.

**Les méthodes**

- *void change\_intensité(Real intensite)* : Changer la valeur de l'intensité de l'abeille ;
- *void déplacer(Int x, Int y)* : Changer la position de l'abeille.

**Class Region**

**Les attributs**

- *Array[] liste\_de\_cellule* : Une liste de cellule ;
- *String type* : Le type de cette région.

**Les méthodes**

- *void ajoute\_cellule()* : Ajouter une cellule dans la région ;
- *void supprime\_cellule(x, y)* : Supprimer la cellule de coordonnée (x,y) dans la région ;
- *void change\_type()* : Changer le type de la région ;
- *void get\_frontiere()* : Trouver la frontière de cette région ;
- *Region[] region\_voisinage()* : Trouver la région voisinage de la région marquée.

## Class Cellule\_abeille

### Les attributs

- *Real intensité* : L'intensité de cette cellule ;
- *Real adéquation* : La valeur d'adéquation ;
- *Real concentration\_p* : La valeur de concentration de la phéromone ;
- *Bool ab* : Existe-t-il un agent dans cette cellule ?
- *int x, int y* : La position dans l'environnement.

### Les méthodes

- *void change\_intensité()* : Changer la valeur d'intensité ;
- *void change\_adéquation()* : Changer la valeur d'adéquation ;
- *void change\_concentration\_p()* : Changer la valeur de concentration de la phéromone.

## Class Environnement\_abeille

### Les attributs

- *Abeille[] abeilles* : Une liste d'abeille dans l'environnement ;
- *Cellule[][] environnement* : Une matrice de cellule,(les pixels au sens de traitement d'image) ;
- *Région[] r\_propag* : Une liste de région de propagation de la phéromone ;
- *Région[] r\_marq* : Une liste de région marquée ;
- *Région[] r\_voisin* : Une liste de région voisinage par rapport à une région marquée.(La longueur de cette liste sera la même comme celle de la région marquée)

### Les méthodes

- *void reset\_env(Int ENV\_SIZE)* : Initialiser l'environnement ;
- *void région\_voisinage()* : Trouver toutes les régions voisines dans l'environnement ;
- *Real load\_environnement(String path, String filename)* : Charger les informations de l'image dans l'environnement ;
- *void région\_marquée()* : Trouver toutes les régions marquées dans l'environnement ;
- *void ajoute\_région(Région région)* : Ajouter la région dans la liste de région ;
- *Abeille[] check\_mort()* : Vérifier si l'agent est mort ou pas sous certain condition ;
- *void ajoute\_abeille()* : Ajouter une abeille dans la liste d'abeille ;
- *void update\_abeilles()* : Mettre à jour les états d'abeille dans l'environnement ;
- *void evaluation\_adéquation(Const Real k = 0.5, alpha = 0.5, beta = 0.5)* : Calculer la valeur d'adéquation pour toutes les cellule dans l'environnement ;
- *void création\_r\_propagation(Const Real seuil)* : Créer les régions de propagation avec une valeur de seuil fixée ;
- *fusionner(Real avgintens)* : Fusionner les régions marquée si besoin ;
- *void evaluation\_conc()* : Calculer la valeur de concentration de phéromone pour toutes les cellule dans l'environnement ;

- *void premier\_déplacement(Real avgintens)* : Le premier déplacement est aléatoire ;
- *int[] choix\_cellule(Abeille agent)* : Choisir l'emplacement d'agent selon la définition de la probabilité de sélection ;
- *void update\_env(Real avgintens)* : Mettre à jour tous les attributs dans l'environnement.

### 4.1.2 Algorithme araignée sociale

#### Object Fil

##### Les attributs

- *Cellule\_araigne end* : le cellule de terminal ;
- *String nom\_groupe* : le groupe auquel l'araignée qui pose ce fil appartient.

#### Class Cellule\_araignee

##### Les attributs

- *Real valeur* : valeur dans la case de l'environnement ;
- *Fil[] fils* : une liste de fil qui commence par cette cellule.

##### Les méthodes

- *Cellule\_araignee[] neigh(Real R)* : générer un ensemble de position (cellules) qui est le voisinage de cette cellule ;
- *Cellule\_araignee[] scut()* : générer un ensemble de positions (cellules) accessibles en suivant un fil ;
- *Cellule\_araignee[] access()* : générer l'union de *neigh* et *scut* ;
- *Int nombre1(Cellule\_araignee b)* : calculer le nombre de fil qui commence par cette cellule et termine par cellule b ;
- *Int nombre2(Cellule\_araignee b, String nom\_groupe)* : calculer le nombre de fil qui commence par cette cellule et termine par cellule b, et posé par le groupe d'araignée "nom\_groupe" ;
- *Void ajout\_fil(Cellule\_araignee b, String nom\_groupe)* : ajouter un fil qui commence par cette cellule et termine par cellule b.

#### Class Araignee

##### Les attributs

- *Real proba\_fil* : la probabilité que l'araignée pose un fil ;
- *Real attract\_self* : le coefficient de l'attraction de même groupe ;
- *Real attract\_other* : le coefficient de l'attraction de différente groupe ;
- *Int x* : position x ;
- *Int y* : position y ;

- *Int R* : rayon pour perception des cases de voisinage ;
- *Cellule\_araignee cellule* : le cellule où l'araignée se trouve ;
- *String groupe\_nom* : le nom du groupe auquel l'araignée appartient.

### Les méthodes

- *Void suivre\_un\_fil()* : l'araignée effectue l'action "suivre un fil existant" ;
- *Cellule\_araignee poser\_fil()* : l'araignée effectue l'action "poser un nouveau fil" ;

### Class Environnement\_araignee

#### Les attributs

- *Int nb\_ligne, nb\_colonne* : nombre de cellule en ligne et en colonne ;
- *Cellule[][] environement* : un environnement constitué par des cellules ;
- *Araignee[] ga* : une liste des araignées qui peuvent appartenir aux différentes groupes.

Puisque le programme sera implémenté en python, nous mettons tous les attributs et toutes les méthodes publiques.

## Chapitre 4. Implémentation

Intelligence en essaim sur l'application de la segmentation des images - Class Diagram

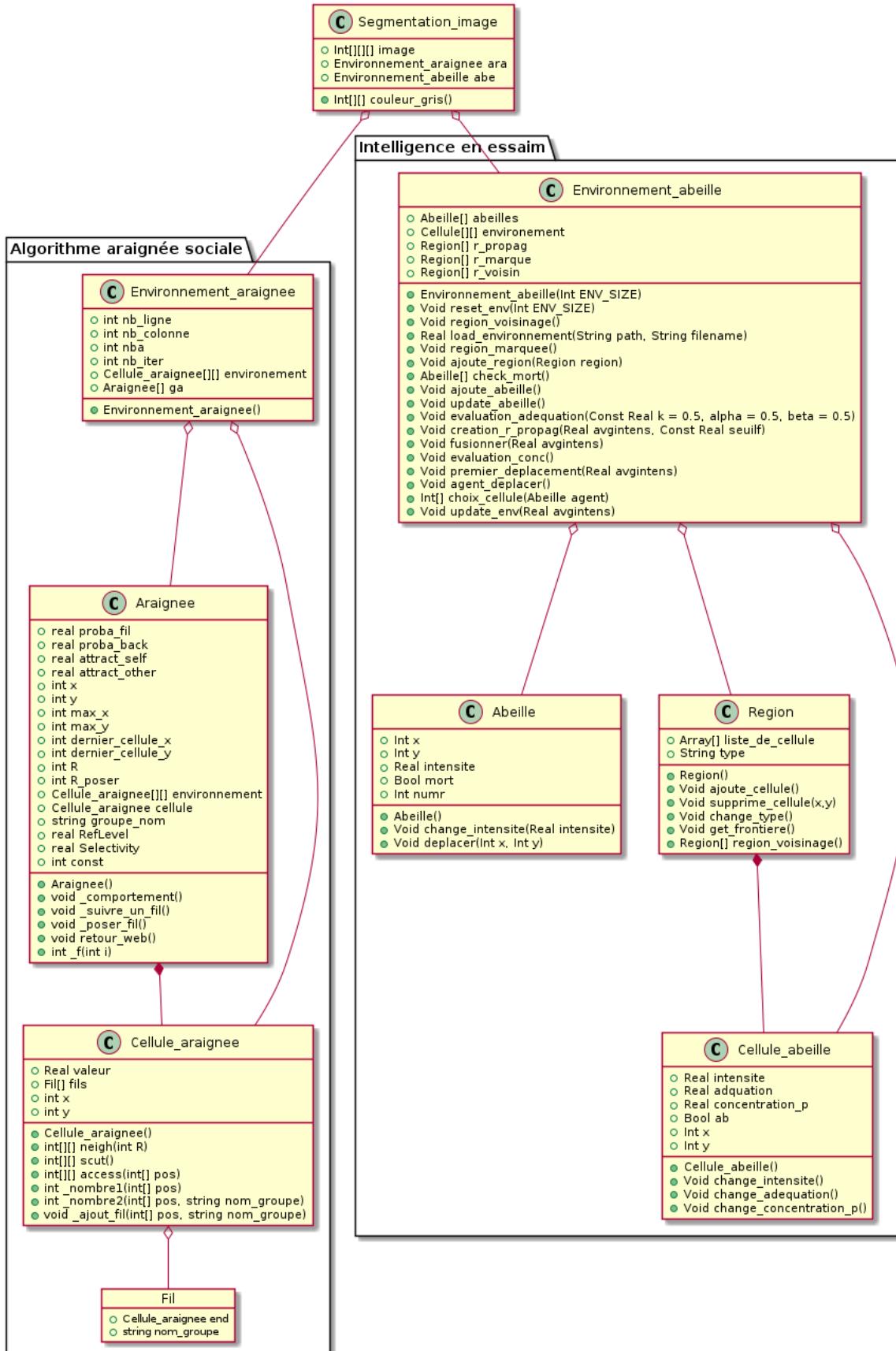


FIGURE 4.1 – UML <sup>17</sup>

# Chapitre 5

## Résultat

### 5.1 Intelligence en essaim

Dans cette section, nous allons expérimenter certains des paramètres nécessaires de l'algorithme. Afin de rendre les performances de l'algorithme plus intuitives dans nos expériences, nous allons expérimenter la photographie 5.1, avec le consentement du photographe. Il s'agit de la photo d'identité d'une femme avec un fond blanc et une large zone de cheveux foncés. Le but est d'identifier la plus grande partie possible des cheveux de la femme.



FIGURE 5.1 – Figure 1

Notre expérience sera divisée en deux parties. Tout d'abord, nous allons étudier le nombre de cycles de l'algorithme, qui affecte le nombre maximum de pas que chaque abeille peut effectuer dans l'algorithme. Deuxièmement, nous étudierons également la taille du paramètre seuil dans l'algorithme, qui affecte la taille de la zone de propagation des phéromones et donc la taille de la zone dans laquelle l'abeille peut se déplacer.

Afin de voir plus clairement la zone identifiée, nous marquons non seulement les résultats de l'algorithme avec des blocs de couleur sur la photographie originale 5.2, mais nous marquons également la zone identifiée en blanc sur une photographie à fond noir, qui est le résultat de notre algorithme dans l'expérience 5.2.

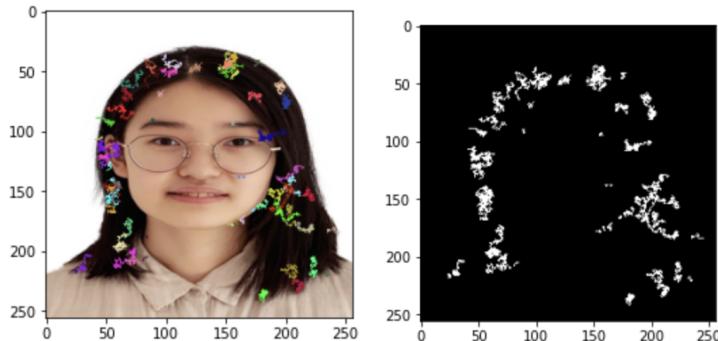


FIGURE 5.2 – Exemples

Dans les deux exemples 5.2, nous pouvons voir que nos expériences sont réalisées dans un environnement de 256\*256 pixels, nous allons donc modifier la taille de l'image originale une fois avant de commencer toutes les expériences.

### 5.1.1 Nombre d'itérations

Dans cette section, nous examinons le nombre de cycles. Nous avons choisi 100, 200, 300, 400 et 500, comme nombre de cycles différents, et leurs temps de calcul ont été enregistrés dans le tableau 5.1.

TABLE 5.1 – Temps de calcul en fonction du nombre d'itérations

Nb d'itérations	100	200	300	400	500
Temps de calcul	43s	5min56s	14min03s	23min	52min33s

Au niveau du temps de calcul, à mesure que l'on augmente le nombre de cycles, le temps de calcul requis par l'algorithme pour obtenir le résultat augmente de façon exponentielle. En effet, plus le nombre de boucles est élevé, plus la longueur de la liste que l'algorithme doit parcourir est importante.

Les figures 5.3 et 5.4 montrent que plus le nombre de cycles augmente, plus la zone de cheveux identifiée par l'algorithme est grande. Nous supposons donc que lorsque le nombre de cycles est suffisamment grand, la zone noire de cheveux sera identifiée dans son intégralité. Au-delà du temps initial, on constate également que l'on peut déjà voir clairement le contour des cheveux à partir du moment où le nombre de cycles atteint trois cents. Par conséquent, afin d'améliorer l'efficacité de la dernière partie de l'expérience, nous avons fixé le nombre de cycles à 300 pour les expériences sur le paramètre seuil.

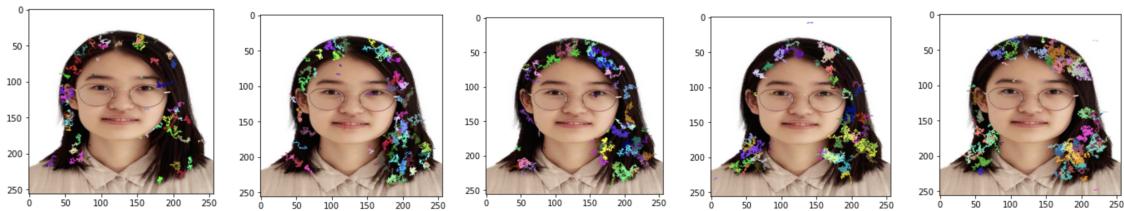


FIGURE 5.3 – Extraction avec différents nombres d’itérations (100, 200, 300, 400, 500)

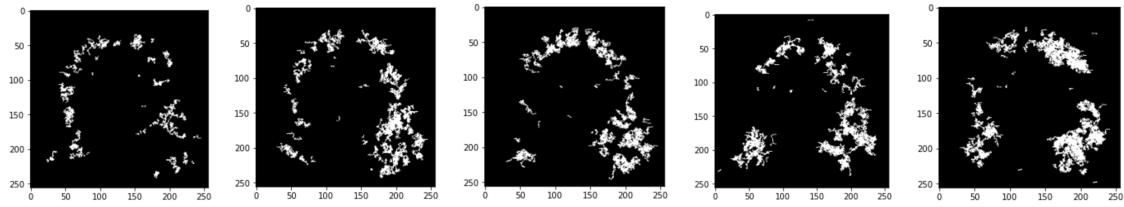


FIGURE 5.4 – Degré d’appartenance de la figure 5.3

### 5.1.2 La valeur de seuil $seuil_f$

Dans cette section, nous étudions les paramètres utilisés dans l’algorithme pour sélectionner la région de propagation de la phéromone. Nous avons choisi 0,1, 0,3, 0,5, 0,8 ,1 comme paramètres expérimentaux et avons enregistré le temps mis par l’algorithme pour produire les résultats dans le tableau 5.2.

TABLE 5.2 – Temps de calcul en fonction de  $seuil_f$

La valeur de $seuil_f$	0.1	0.3	0.5	0.8	1
Temps de calcul	1min19s	8min41s	10min23s	14min03s	14min55s

Nous pouvons constater que plus la valeur de ce paramètre augmente, plus l’algorithme met du temps à produire ses résultats. Ensuite, nous pouvons comparer les surfaces des zones marquées des images.

Comme nous pouvons le voir dans les figures 5.5 et 5.6, lorsque nous augmentons la valeur de ce paramètre, la zone marquée augmente, mais lorsque nous fixons la valeur de ce paramètre à 1, nous pouvons voir que l’algorithme marque certaines zones non poilues, un phénomène que nous ne voulons pas voir, nous pensons donc que 0,8 est la meilleure valeur pour ce paramètre.

## 5.2 Algorithme des araignées sociales

Nous avons appliqué l’algorithme des araignées sociales sur la fig. 5.1.

## Chapitre 5. Résultat

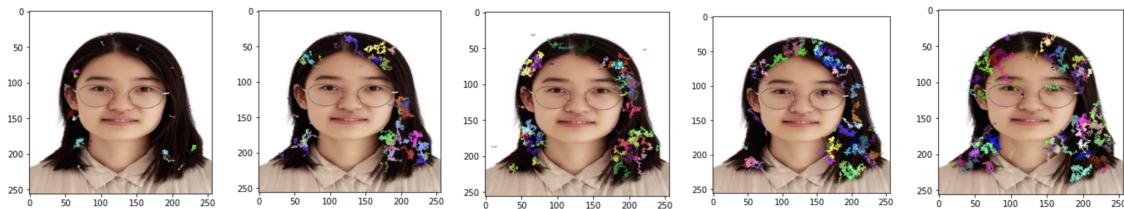


FIGURE 5.5 – Extraction avec différents valeurs de  $seuil_f$  (0.1, 0.3, 0.5, 0.8, 1.0)

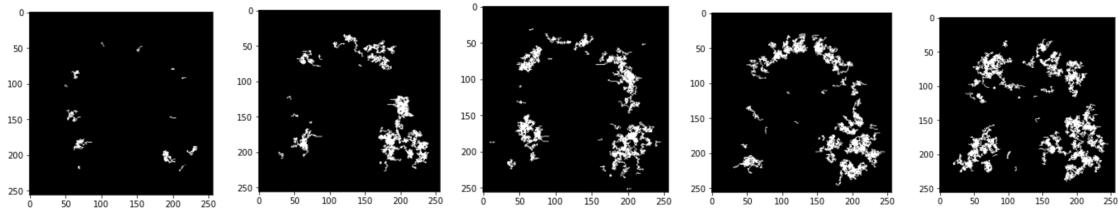


FIGURE 5.6 – Degré d'appartenance de la figure 5.5

Nous souhaitons que cet algorithme peut bien distinguer la partie cheveux avec une couleur gris plus basse que d'autres parties. Le résultat du premier lancement est montré dans la figure 5.7.

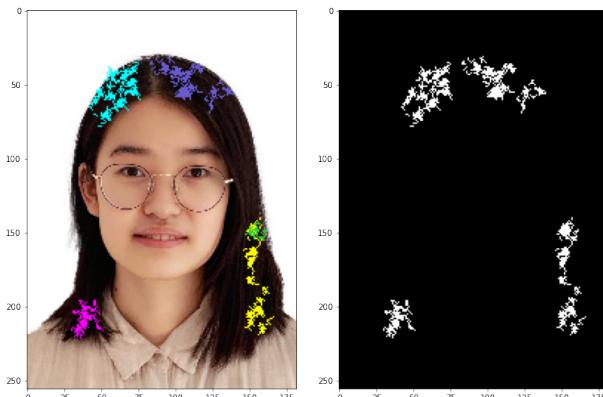


FIGURE 5.7 – Premier essai

Dans la partie suivante, nous allons relier les paramètres individuels initiaux aux résultats collectifs finaux afin de trouver des heuristiques pour estimer les paramètres nécessaires à l'extraction d'une région spécifique. Nous nous concentrerons sur la relation entre les paramètres du modèle et les résultats observés. Si l'on considère que tous les agents ont les mêmes caractéristiques, un modèle est décrit par trois paramètres : les variables *proba\_back* et *selectivity*.

Les deux premiers paramètres régissent les déplacements des agents et donc le processus d'exploration. Les deux derniers sont liés à la sélection des pixels déterminant ainsi la

pertinence des régions extraites. Sauf ces quatre paramètres, nous nous intéressons aussi au nombre d’itérations afin de limiter le temps de calcul.

### 5.2.1 Nombre d’itérations

Nous étudions tout d’abord l’importance de nombre d’itérations. Le programme avec 1000 itérations développe peu de régions par rapport aux autres nombre d’itérations. La trajectoire de chaque agent se concentre à une région, et ne croise pas les autres. Ainsi, 1000 cycles ne permettent pas de parcourir toutes régions cheveux et il faut ajouter plus de cycles.

Nous constatons que dans la figure 5.8 avec 10000 cycles, l’agent rose traverse les cheveux à gauche de bas en haut, ce qui implique que 10000 cycles sont suffisantes pour aller plus loin. De plus, le temps de calcul du 10000 cycles ne prend que de quelques secondes par rapport à l’exécution de 20000 cycles prend 3 minutes.

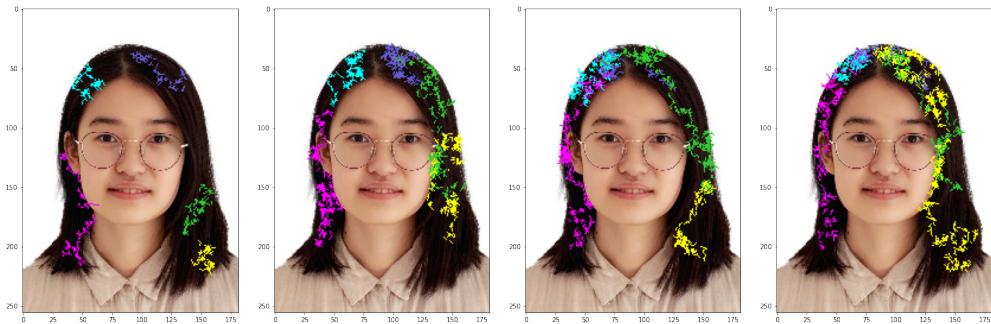


FIGURE 5.8 – Extraction avec différents nombre d’itérations (1000, 5000, 10000, 20000 nombre d’itérations)

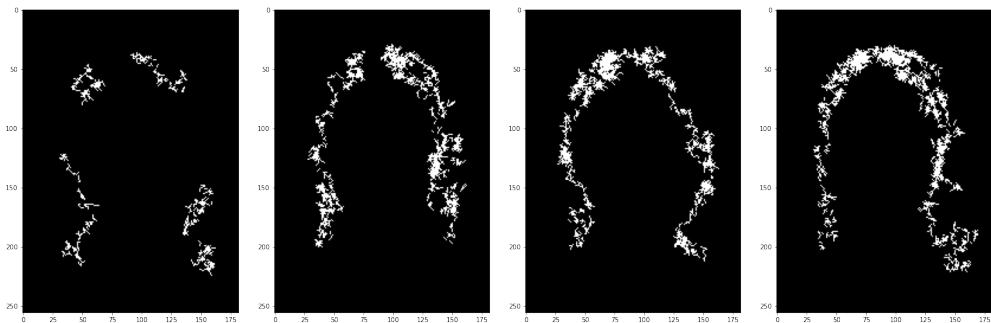


FIGURE 5.9 – Degré d’appartenance de la figure 5.8

### 5.2.2 Probabilité de retour au toile

L’importance de *proba\_back* est illustrée par les figures 5.10 et 5.11. Si la probabilité est nulle, les agents pourraient atteindre une autre région, puis tisser des fils reliant ces deux

régions. C'est le cas de la figure 5.10 où une toile a été tissée les montures de lunettes. Cependant, la *proba\_back* ne doit pas être trop élevée, car une *proba\_back* faible permet aux agents de traverser de petites zones bruitées pour poursuivre les processus de sélection.

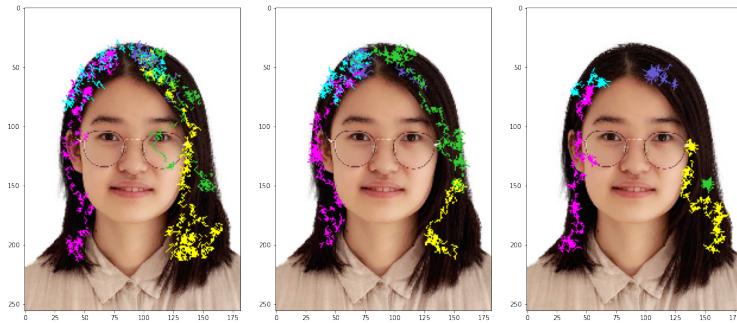


FIGURE 5.10 – Extraction avec différentes probabilités de retour au toile (0, 0.2, 0.4)

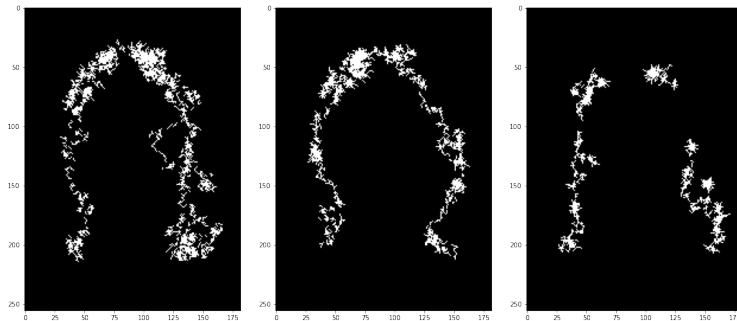


FIGURE 5.11 – Degré d'appartenance de la figure 5.10

### 5.2.3 Sélectivité

Sur la *sélectivité*, si la région que nous allons extraire est bien séparé par le niveau gris, la *selectivity* doit être faible pour tenir compte des petites fluctuations de niveau de gris dans la région. Au contraire, si la région n'est pas bien détachée de l'image, la *selectivity* doit être élevé pour extraire les limites attendues. Dans notre cas, puisque le niveau de gris des cheveux est bien distingué de la visage et du fond, la variation de la *selectivity* a peu d'influence sur le résultat, comme montré dans les figures 5.12 et 5.13.

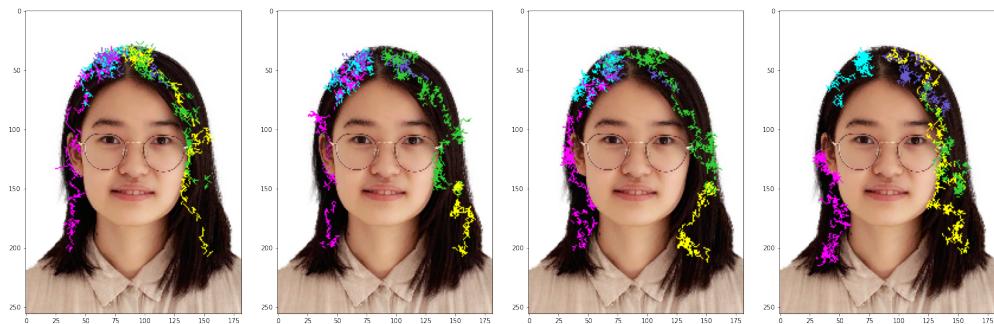


FIGURE 5.12 – Extraction avec différentes sélectivités (0.1, 0.5, 1, 2)

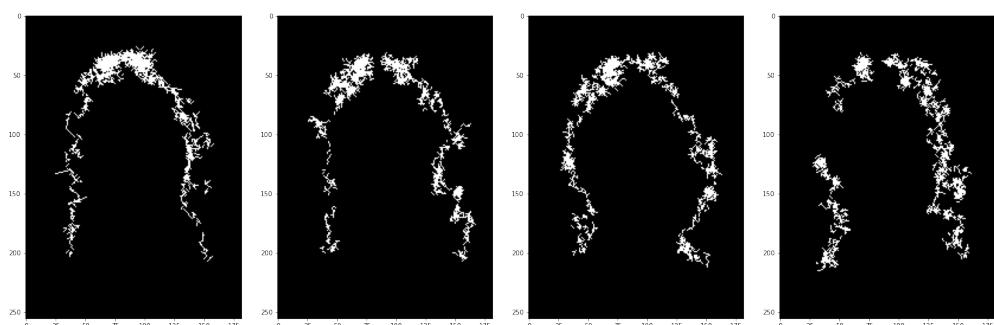


FIGURE 5.13 – Degré d'appartenance de la figure 5.12

# **Chapitre 6**

## **Conclusion**

Dans ce projet, nous avons étudié des algorithmes artificiels et implémenté deux des plus courants. En appliquant ces deux algorithmes au domaine de la segmentation d'images, on reconnaît clairement l'importance de ce type d'algorithme dans le domaine de l'intelligence artificielle.

Dans notre exemple, on voit clairement les capacités de ces deux algorithmes, bien sûr, dans ce domaine, on a encore du chemin à faire.

À partir du niveau de notre projet, nous devons encore étudier plus avant les détails de l'algorithme, la structure de données utilisée dans le code et les réglages de paramètres importants, afin d'améliorer les performances de l'algorithme.

Du point de vue de l'ensemble du domaine de l'intelligence artificielle, il reste encore de nombreuses parties à améliorer dans l'application des algorithmes bioniques au domaine de la découpe d'images, par exemple l'introduction d'autres modèles biologiques, etc.

# Bibliographie

- [1] L. DJEROU and M. BATOUCHÉ, “Résolution collective du problème de segmentation,”
- [2] D. Safia and M. Batouche, “Une approche bio-mimétique pour la segmentation d’images inspiration des araignées sociales.,” 01 2009.
- [3] C. Bourjot, V. Chevrier, A. Bernard, and B. Krafft, “Coordination par le biais de l’environnement : une approche biologique,” in *Journées Francophones sur l’Intelligence Artificielle Distribuée et les Systèmes Multi-agents* (P. M. M.P. Gleizes, ed.), (St Gilles les bains, La Réunion), pp. 237–250, Hermes, 1999. Colloque avec actes et comité de lecture.