IMT Mines Alès
École Mines-Télécom

AntAlgorithm

Cahier de recettage

Rédacteurs :
Océane MONGES

Date : 20 janv 2025 - Version V01

Visa :

IMT Mines Alès
École Mines-Télécom

SOMMAIRE

SOMMAIRE **1**
TABLES DES MODIFICATIONS **2**
1. INTRODUCTION **2**
1.1. Contexte 2
1.2. Objectifs du projet 2
1.3. Objectif du Cahier de Recettage 2
2. ORGANISATION DES TESTS **3**
3. SCÉNARIO DE TESTS **3**

IMT Mines Alès
École Mines-Télécom

TABLES DES MODIFICATIONS

Révision	Date	Page ou §	Objet de la révision
V1	20 janv 2025	Toutes	

1. INTRODUCTION

Le cahier de recettage définit les critères de validation pour le projet "AntAlgorithm", en s'assurant que les fonctionnalités et comportements spécifiés dans le cahier des charges sont conformes aux attentes.

1.1. Contexte

Dans le cadre d'un projet académique, une équipe de deux personnes ont développé un système multi-agents simulant une colonie de fourmis, en s'appuyant sur l'algorithme d'optimisation par colonie de fourmis (Ant Colony Optimization, ACO). Cet algorithme, inspiré du comportement naturel des fourmis, permet de résoudre des problèmes complexes d'optimisation en exploitant des mécanismes tels que le dépôt et l'évaporation de phéromones pour identifier des chemins optimaux.
Le projet vise à modéliser ces interactions pour aboutir à une solution efficace et collaborative, reflétant les dynamiques observées dans les colonies de fourmis réelles.

1.2. Objectifs du projet

- Développer un système multi-agents permettant à des agents (fourmis) de trouver et d'apporter des ressources (nourriture) à leur point de départ (colonie).
- Implémenter une simulation où les agents interagissent avec un environnement représenté par une grille composée de différents types de cellules.
- Utiliser l'algorithme de colonies de fourmis pour optimiser les trajets des agents en se basant sur des traces de phéromones.

1.3. Objectif du Cahier de Recettage

Le but du cahier de recettage est de fournir une méthode structurée pour tester et valider :

- Les fonctionnalités principales de la simulation.
- La conformité de l'interface utilisateur.
- Les performances du système selon les spécifications définies.

2. ORGANISATION DES TESTS

Les tests du projet "AntAlgorithm" seront menés selon une approche méthodique et structurée, divisée en quatre phases principales, chacune ciblant des aspects spécifiques de la solution :

Tests de Préparation	Cette phase vise à vérifier l'installation correcte et la configuration initiale du projet. Elle inclut la validation de l'environnement de développement, notamment le serveur local (Node.js ou autre), ainsi que le bon fonctionnement des dépendances et des modules JavaScript. L'objectif est de garantir que tous les développeurs disposent d'un environnement uniforme, prêt à exécuter la simulation sans erreurs techniques. On vérifiera également que les fichiers HTML, CSS et JavaScript sont correctement reliés.
Tests Fonctionnels	Ces tests se concentrent sur la validation des fonctionnalités principales décrites dans le cahier des charges. Cela inclut la vérification de la logique de déplacement des agents, la gestion des phéromones, le retour à la colonie, et les interactions avec les différents types de cellules (obstacles, objectifs, libres). Cette phase permet de s'assurer que les fonctionnalités s'exécutent selon les règles définies, sans bugs, et que chaque agent respecte les contraintes de comportement établies.
Tests de l'Interface Utilisateur	L'objectif ici est de valider les interactions utilisateur et le rendu graphique de la simulation. Les contrôles de simulation (démarrage, pause, réinitialisation) seront testés pour garantir leur bon fonctionnement. On évaluera également la clarté et l'intuitivité de l'interface : les types de cellules doivent être facilement identifiables, les phéromones doivent être représentées de manière visuelle explicite (via un gradient de couleurs), et les déplacements des fourmis doivent être animés de façon fluide. Une attention particulière sera portée aux informations affichées en temps réel, comme les niveaux de phéromones et l'état des ressources.
Tests de Robustesse et de Performance	Cette dernière phase évalue la stabilité du système sous diverses charges. Elle inclut des scénarios où un grand nombre d'agents évoluent simultanément sur une grille complexe. On vérifie que le système reste fonctionnel sans ralentissements ou plantages, et que la gestion des phéromones reste cohérente même en cas d'activité intense. Cette phase identifie également les limites potentielles de l'implémentation, notamment en termes de consommation de ressources et de temps de calcul. Chaque phase sera documentée avec soin, et les résultats des tests seront analysés pour identifier les points à corriger ou améliorer. Cette organisation garantit une couverture complète et méthodique, alignée sur les exigences du projet.

Chaque phase sera documentée avec soin, et les résultats des tests seront analysés pour identifier les points à corriger ou améliorer. Cette organisation garantit une couverture complète et méthodique, alignée sur les exigences du projet.

3. SCÉNARIO DE TESTS

ID	Description	
PREP-Scénario de tests de préparation		
PREP-001	Objectifs	Vérifier le bon fonctionnement du serveur local
	Prérequis	Serveur local installé (Node.js, Live Server, etc.)
	Etapes	1. Lancer le serveur local 2. Accéder à l'URL localhost 3. Vérifier le chargement de index.html
	Résultat attendu:	- Le serveur démarre sans erreur - La page index.html s'affiche correctement - Pas d'erreurs dans la console
	Validation	OUI
PREP-002	Objectifs	Vérifier le chargement des modules ES6
	Prérequis	Serveur local fonctionnel
	Etapes	1. Ouvrir la console du navigateur 2. Vérifier les imports dans index.js 3. Vérifier le chargement de: - ControlPanel.controller.js - Canvas.controller.js - GameEngine.js
	Résultat attendu:	- Aucune erreur de type "Cannot use import statement outside a module" - Tous les modules sont correctement chargés
	Validation	OUI
PREP-003	Objectifs	Vérifier le chargement des assets graphiques
	Prérequis	ImageLoader configuré
	Etapes	1. Vérifier le chargement des images dans ImageLoader: - foodAndColony.png - grass.png - cellules.png - tree.png - ant.png 2. Contrôler les propriétés de chaque image
	Résultat attendu:	- Toutes les images sont chargées - Les dimensions sont correctes - Pas d'erreurs 404 dans la console
	Validation	OUI

PREP-004	Objectifs	Vérifier le chargement des sons
	Prérequis	Fichiers audio présents
	Etapes	1. Vérifier la présence des fichiers audio: - crunch.mp3 - pop.mp3 2. Tester le chargement dans les classes concernées
	Résultat attendu:	- Les fichiers audio sont accessibles - Pas d'erreurs de chargement
	Validation	OUI
PREP-005	Objectifs	Vérifier la structure DOM
	Prérequis	index.html chargé
	Etapes	1. Vérifier la présence des éléments: - Canvas #game - Bouton #control - Bouton #phéromones - Span #chrono 2. Vérifier les dimensions du canvas
	Résultat attendu:	- Tous les éléments sont présents - Le canvas a les bonnes dimensions (90% de la hauteur de l'écran)
	Validation	OUI
PREP-006	Objectifs	Vérifier l'application des styles
	Prérequis	index.css chargé
	Etapes	1. Vérifier l'application des styles: - Fond d'écran avec motif - Boutons stylisés - Chronomètre visible 2. Tester la réactivité
	Résultat attendu:	- Styles appliqués correctement - Interface responsive (non applicable dans notre cadre) - Pas de débordements
	Validation	OUI
PREP-007	Objectifs	Vérifier les performances de base du canvas
	Prérequis	Canvas initialisé
	Etapes	1. Mesurer le FPS initial 2. Vérifier la mémoire utilisée 3. Tester avec différentes tailles de grille

	Résultat attendu:	- FPS stable à 60 - Utilisation mémoire raisonnable - Pas de fuites mémoire
	Validation	OUI
FONC-Scénario de tests fonctionnels		
FUNC-001	Objectifs	Vérifier la génération correcte de la grille
	Prérequis	Application initialisée
	Etapes	1. Lancer une nouvelle partie 2. Vérifier la génération de la grille 20x20 3. Contrôler les bordures (arbres)
	Résultat attendu:	- Grille 20x20 générée - Bordures composées uniquement d'arbres - Distribution aléatoire correcte des cellules (40% arbres)
	Validation	OUI
FUNC-002	Objectifs	Vérifier le placement correct des éléments spéciaux
	Prérequis	Grille générée
	Etapes	1. Vérifier la présence: - 1 fourmilière (ANTHILL) - 2 à 4 sources de nourriture (FOOD) - 2 obstacles (OBSTACLE) 2. Vérifier l'accessibilité de chaque élément
	Résultat attendu:	- Éléments placés uniquement sur des cases libres - Chemins accessibles entre fourmilière et nourriture - Quantité initiale de nourriture = 1.0
	Validation	OUI
FUNC-003	Objectifs	Vérifier la création et limitation des fourmis
	Prérequis	Simulation en cours
	Etapes	1. Démarrer la simulation 2. Observer la génération progressive des fourmis 3. Attendre jusqu'à atteindre la limite
	Résultat attendu:	- Génération aléatoire (2% de chance par frame) - Maximum 30 fourmis - Toutes les fourmis partent de la fourmilière
	Validation	OUI
FUNC-004	Objectifs	Vérifier le comportement de déplacement

	Prérequis	Fourmis actives
	Etapes	1. Observer les déplacements: - Mouvement case par case - Évitement des obstacles - Respect des bordures 2. Vérifier la perception des cases adjacentes
	Résultat attendu:	- Déplacement fluide - Pas de traversée d'obstacles - Vision limitée aux cases adjacentes
	Validation	TEST NON IMPLÉMENTÉ
FUNC-005	Objectifs	Vérifier le système de dépôt de phéromones
	Prérequis	Fourmi ayant trouvé de la nourriture
	Etapes	1. Suivre une fourmi jusqu'à la nourriture 2. Observer le retour à la fourmilière 3. Vérifier le dépôt de phéromones
	Résultat attendu:	- Dépôt de 0.3 unités par case - Visualisation correcte des phéromones - Pas de dépôt sur obstacles/arbres
	Validation	TEST NON IMPLÉMENTÉ
FUNC-006	Objectifs	Vérifier l'évaporation des phéromones
	Prérequis	Phéromones déposées
	Etapes	1. Observer une piste de phéromones 2. Mesurer la diminution au fil du temps 3. Vérifier le taux d'évaporation
	Résultat attendu:	- Taux d'évaporation de 0.005 - Évaporation progressive visible - Disparition complète après inactivité
	Validation	TEST NON IMPLÉMENTÉ
FUNC-007	Objectifs	Vérifier le processus de collecte
	Prérequis	Source de nourriture accessible
	Etapes	1. Observer une fourmi trouvant de la nourriture 2. Vérifier la collecte (0.05 unité) 3. Contrôler la diminution de la source
	Résultat attendu:	- Collecte correcte de 0.05 unité - Mise à jour visuelle de la quantité - Son de collecte joué

	Validation	TEST NON IMPLÉMENTÉ
FUNC-008	Objectifs	Vérifier l'épuisement et le renouvellement
	Prérequis	Source de nourriture active
	Etapes	1. Observer l'épuisement d'une source 2. Vérifier la transformation en case libre 3. Contrôler l'apparition d'une nouvelle source
	Résultat attendu:	- Disparition à 0 unité - Génération nouvelle source aléatoire - Son de placement joué
	Validation	TEST NON IMPLÉMENTÉ
UI-Scénario de tests de l'interface graphique		
UI-001	Objectifs	Vérifier l'affichage initial de l'interface
	Prérequis	Application initialisée
	Etapes	1. Ouvrir l'application 2. Vérifier les éléments visuels: - Canvas centré - Chronomètre (00:00:00) - Bouton "Start" - Bouton "Phéromones" - Arrière-plan avec motif d'arbres
	Résultat attendu:	- Tous les éléments sont visibles - Alignement correct - Police et couleurs conformes - Pas de chevauchement d'éléments Cell
	Validation	OUI
UI-002	Objectifs	Vérifier la représentation visuelle des cellules
	Prérequis	Grille générée
	Etapes	1. Vérifier le rendu de chaque type de cellule: - Herbe (FLOOR) - Arbre (TREE) - Fourmilière (ANTHILL) - Nourriture (FOOD) - Obstacle (OBSTACLE) 2. Contrôler les superpositions d'images
	Résultat attendu:	- Images chargées correctement - Taille proportionnelle des sprites - Distinction claire entre types - Pas d'artefacts graphiques

	Validation	OUI
UI-003	Objectifs	Vérifier le rendu et l'animation des fourmis
	Prérequis	Simulation en cours
	Etapes	1. Observer le mouvement des fourmis: - Fluidité du déplacement - Rotation selon direction - Superposition avec cellules 2. Vérifier différentes vitesses de simulation
	Résultat attendu:	- Animation fluide (60 FPS) - Rotations sans saccades - Pas de traînées graphiques - Taille proportionnelle des fourmis
	Validation	OUI
UI-004	Objectifs	Vérifier la visualisation des phéromones
	Prérequis	Phéromones déposées
	Etapes	1. Activer/désactiver l'affichage des phéromones 2. Vérifier les deux modes d'affichage: - Valeurs numériques - Cercles colorés 3. Observer les transitions d'intensité
	Résultat attendu:	- Toggle fonctionnel - Lisibilité des valeurs - Gradient de couleur cohérent - Mise à jour en temps réel
	Validation	OUI
UI-005	Objectifs	Vérifier l'affichage et la précision du chronomètre
	Prérequis	Simulation active
	Etapes	1. Observer le format d'affichage (HH:MM:SS:MS) 2. Vérifier la mise à jour 3. Tester pause/reprise
	Résultat attendu:	- Format correct - Mise à jour fluide - Précision du timing - Persistance en pause
	Validation	OUI
UI-006	Objectifs	Vérifier l'interaction avec les boutons

	Prérequis	Interface chargée
	Etapes	1. Tester les états des boutons: - Hover - Click - État actif/inactif 2. Vérifier les transitions d'état
	Résultat attendu:	- Feedback visuel au hover - Changement d'état visible - Texte lisible - Transitions fluides
	Validation	OUI
UI-007	Objectifs	Vérifier les retours sonores
	Prérequis	Sons chargés
	Etapes	1. Vérifier les sons: - Collecte de nourriture (crunch.mp3) - Nouvelle source (pop.mp3) 2. Tester le timing et le volume
	Résultat attendu:	- Sons joués au bon moment - Volume approprié - Pas de latence - Pas de superposition gênante
	Validation	TEST NON IMPLÉMENTÉ
PERF-Scénario de tests de performances		
PERF-001	Objectifs	Évaluer les performances de rendu
	Prérequis	Application en état de fonctionnement normal
	Etapes	1. Mesurer les FPS: - Au démarrage - Avec 10 fourmis - Avec 30 fourmis (maximum) 2. Monitorer pendant 30 minutes
	Résultat attendu:	- FPS stable à 60 - Pas de chute en dessous de 30 FPS - Temps de rendu < 16ms par frame - Utilisation CPU < 50%
	Validation	OUI
PERF-002	Objectifs	Vérifier l'utilisation de la mémoire
	Prérequis	DevTools ouvert

	Etapes	1. Mesurer la consommation: - Au démarrage - Après 10 minutes - Après 30 minutes 2. Forcer le garbage collector 3. Vérifier les fuites mémoire
	Résultat attendu:	- Utilisation mémoire stable - Pas de croissance continue - Pas de fuites mémoire - Pic mémoire < 200MB
	Validation	OUI
PERF-003	Objectifs	Tester le comportement sous charge maximale
	Prérequis	Simulation active
	Etapes	1. Atteindre 30 fourmis 2. Activer l'affichage des phéromones 3. Créer maximum de traces de phéromones 4. Maintenir pendant 1 heure
	Résultat attendu:	- Performance stable - Pas de crash - Réactivité UI maintenue - Comportement fourmis cohérent
	Validation	TEST NON IMPLÉMENTÉ
PERF-004	Objectifs	Vérifier la gestion des calculs intensifs
	Prérequis	Environnement complexe généré
	Etapes	1. Tester avec configuration maximale: - 30 fourmis actives - Multiples sources de nourriture - Maximum de phéromones 2. Mesurer temps de calcul: - Pathfinding - Évaporation phéromones - Calculs probabilités
	Résultat attendu:	- Temps de calcul < 5ms par itération - Pas de blocage UI - Précision des calculs maintenue
	Validation	TEST NON IMPLÉMENTÉ
PERF-005	Objectifs	Évaluer les performances avec différentes tailles de grille
	Prérequis	Possibilité de modifier la taille de la grille

	Etapes	1. Tester avec grilles: - Petite (10x10) - Standard (20x20) - Grande (40x40) - Très grande (100x100) 2. Pour chaque taille: - Mesurer FPS - Monitorer CPU - Vérifier temps de génération - Tester avec 30 fourmis
	Résultat attendu:	- 10x10: FPS stable 60, génération < 100ms - 20x20: FPS stable 60, génération < 200ms - 40x40: FPS > 45, génération < 500ms - 100x100: FPS > 30, génération < 1000ms
	Validation	TEST NON IMPLÉMENTÉ
ROB-Scénario de tests de robustesse		
ROB-001	Objectifs	Vérifier la gestion des erreurs
	Prérequis	Accès au code source
	Etapes	1. Tester les cas d'erreur: - Chargement images échoué - Sons non disponibles - Coordonnées invalides 2. Vérifier les mécanismes de fallback
	Résultat attendu:	- Messages d'erreur appropriés - Pas de crash application - Récupération gracieuse - Logging des erreurs
	Validation	TEST NON IMPLÉMENTÉ
ROB-002	Objectifs	Vérifier la stabilité sur longue période
	Prérequis	Environnement stable
	Etapes	1. Exécuter simulation pendant 24h 2. Monitorer: - Utilisation ressources - Comportement fourmis - État phéromones 3. Vérifier intégrité données
	Résultat attendu:	- Pas de dégradation performance - Cohérence des données - Stabilité du système - Pas d'accumulation d'erreurs

	Validation	TEST NON IMPLÉMENTÉ
ROB-003	Objectifs	Vérifier le comportement aux limites
	Prérequis	Accès aux paramètres système
	Etapes	1. Tester les limites: - Quantité maximale phéromones - Nombre maximum fourmis - Taille grille maximale 2. Vérifier dépassements
	Résultat attendu:	- Gestion correcte des limites - Pas de débordement - Messages d'avertissement - Comportement défini
	Validation	TEST NON IMPLÉMENTÉ
ROB-004	Objectifs	Vérifier la récupération après erreur
	Prérequis	Simulation en cours
	Etapes	1. Provoquer des erreurs: - Perte connexion - Erreur JavaScript - Surcharge mémoire 2. Observer récupération
	Résultat attendu:	- Reprise automatique - Conservation état - Logs d'erreur - Notification utilisateur
	Validation	TEST NON IMPLÉMENTÉ
ROB-005	Objectifs	Vérifier la compatibilité navigateurs
	Prérequis	Différents navigateurs installés
	Etapes	1. Tester sur: - Chrome - Firefox - Edge 2. Vérifier performance relative
	Résultat attendu:	- Comportement cohérent - Performance similaire - Rendu identique - Fonctionnalités préservées
	Validation	OUI