# Consensus au sein d'un essaim de robots

Martin BICHE
TELECOM Nancy – Promotion 2026
martin.biche@telecomnancy.eu

Sylvie SIDLER
TELECOM Nancy – Promotion 2026
sylvie.sidler@telecomnancy.eu

Zoé VERNICOS
TELECOM Nancy – Promotion 2026
zoe.vernicos@telecomnancy.eu

Amine BOUMAZA
Enseignant-chercheur au LORIA
amine.boumaza@loria.fr

***Abstract*—Dans le domaine de la robotique en essaim, l'émergence de comportements collectifs cohérents à partir d'interactions locales représente un défi majeur. Ce projet se concentre sur l'implémentation et l'analyse du Naming Game, un modèle d'émergence de consensus linguistique, au sein d'un groupe de robots Kilobots. Notre objectif est d'évaluer les dynamiques temporelles nécessaires à un groupe de N robots pour atteindre un accord global.**

**Nous avons adapté le Naming Game aux contraintes matérielles des Kilobots et développé une approche méthodologique combinant trois axes : la programmation des robots, l'utilisation du simulateur Kilombo, et l'analyse d'images pour le suivi des interactions. Les résultats montrent que la structure du réseau d'interaction impacte significativement le temps de convergence, les configurations se rapprochant d'un graphe complet atteignant un consensus plus rapidement.**

**Cet article présente les différentes étapes du projet, les principaux défis rencontrés, ainsi que les solutions apportées. Les résultats obtenus montrent qu'un robot aussi simple que le Kilobot peut jouer un rôle significatif dans la recherche en robotique.**

**Mots clés : robotique en essaim, analyse d'image, diffusion de l'information, Naming Game**

## I. Introduction

Dans le domaine des systèmes distribués, la capacité d'un ensemble d'agents autonomes à parvenir à un consensus sans supervision centrale est une problématique fondamentale, avec des applications dans les réseaux de capteurs, les systèmes multi-robots, ou encore les algorithmes de coordination collective. Parmi les nombreux modèles proposés pour étudier ce phénomène, le Naming Game constitue un cadre théorique simple mais puissant pour analyser la dynamique de formation d'un consensus linguistique au sein d'une population d'agents interagissant localement.

Ce projet s'inscrit dans cette perspective et s'intéresse à l'émergence d'un consensus dans une population de robots Kilobots (Figure 1) en mettant en œuvre le Naming Game. Plus précisément, notre objectif est d'évaluer combien de temps il faut à un groupe de N robots pour parvenir à un consensus, c'est-à-dire pour qu'ils adoptent tous un même mot à l'issue d'interactions locales successives.

Pour cela, nous avons implémenté une version du Naming Game adaptée aux contraintes matérielles et de communication des Kilobots. Afin de pouvoir étudier le comportement du système à différentes échelles (en nombre de robots, en topologie de communication, etc.), nous nous sommes appuyés sur Kilombo un simulateur open-source dédié à ces robots, que nous avons adapté pour intégrer notre protocole.

La suite de cet article est organisée comme suit : nous présentons d'abord un état de l'art sur le Naming Game et ses applications robotiques. Nous décrivons ensuite la méthodologie employée pour l'implémentation et l'expérimentation dans Kilombo. Nous discutons ensuite les résultats obtenus en fonction de différents paramètres. Enfin, nous concluons et ouvrons sur des perspectives d'amélioration ou d'application future.

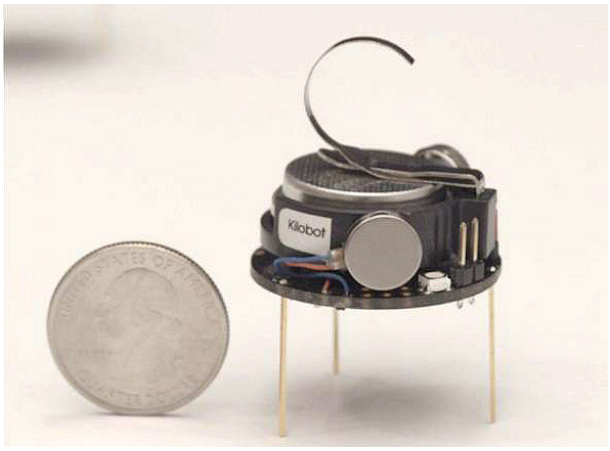


Fig. 1. Kilobot (Josh Cassidy/KQED Science).

## II. État de l'art

L'émergence de comportements coordonnés et de consensus au sein des systèmes multi-agents décentralisés, particulièrement en robotique en essaim, constitue un domaine de recherche foisonnant. Ce projet s'inscrit dans cette lignée en

explorant l'implémentation et la dynamique du consensus via le Naming Game (jeu de dénomination) sur un essaim de robots Kilobot.

### A. Robotique en Essaim et Intelligence Distribuée

La robotique en essaim s'inspire de systèmes biologiques collectifs, tels que les colonies d'insectes, pour concevoir des systèmes composés d'un grand nombre de robots relativement simples et peu coûteux [5]. Individuellement, ces robots possèdent des capacités limitées, mais leurs interactions locales permettent l'émergence de comportements collectifs complexes et robustes, relevant de l'intelligence distribuée [9]. L'un des défis majeurs de cette approche réside dans la conception de mécanismes de coordination décentralisés permettant à l'essaim d'accomplir des tâches et de parvenir à un accord sans autorité centrale. Les Kilobots, par leur faible coût et leur simplicité, se sont imposés comme un outil privilégié pour l'étude expérimentale de ces phénomènes à grande échelle [8].

### B. Le Naming Game comme Modèle d'Émergence de Consensus

Le Naming Game (NG) est un modèle paradigmatique pour l'étude de l'émergence de conventions partagées, telles qu'un vocabulaire commun, au sein d'une population d'agents [3]. Proposé initialement par Steels, il simule comment des agents, par le biais d'interactions dyadiques locales, peuvent converger vers un lexique partagé sans supervision externe. Dans sa version minimale (Minimal Naming Game, MNG), un agent "locuteur" (speaker) propose un nom pour un objet (ou concept) à un agent "auditeur" (listener ou hearer selon les versions). Si l'auditeur connaît déjà ce nom et l'associe au même objet, l'interaction est un succès, et les deux agents peuvent renforcer cette association (par exemple, en ne gardant que ce nom dans leur inventaire pour cet objet). En cas d'échec (l'auditeur ne connaît pas le nom ou l'associe à autre chose), l'auditeur peut ajouter ce nouveau nom à son inventaire. Ce processus itératif (algorithme 2) mène, sous certaines conditions, à un consensus global où tous les agents utilisent le même nom pour le même objet [4].

Les dynamiques du NG ont été extensivement étudiées d'un point de vue théorique et par des simulations. Baronchelli, Loreto et Steels (2008) [1] ont fourni une analyse quantitative détaillée, examinant l'influence de la taille de la population sur des métriques clés telles que le temps de convergence et les besoins en mémoire des agents. Ils ont mis en évidence la capacité du modèle à atteindre une coordination globale via des interactions purement locales et des mécanismes simples. Des travaux ultérieurs, comme ceux de Baronchelli (2016) [4], ont exploré le MNG sous l'angle des systèmes complexes et de la physique statistique, analysant notamment l'impact de la topologie du réseau d'interaction sur la dynamique de convergence.

### C. Implémentation du Naming Game sur des Essaims de Robots

La transition des modèles théoriques du NG vers des plateformes robotiques physiques soulève des défis spécifiques. Trianni et al. (2016) [2] ont étudié l'émergence du consensus en implémentant le NG sur des essaims de robots Kilobot. Leurs travaux ont mis en lumière l'impact de l'incarnation physique et des interactions dans le monde réel. Ils ont observé que l'exécution concurrente des jeux par plusieurs robots, les interférences physiques (collisions) affectant la mobilité, et la densité des robots influencent crucialement la dynamique du consensus. Bien que les interférences physiques puissent ralentir la convergence par rapport aux simulations idéalisées, elles peuvent aussi paradoxalement réduire la charge cognitive des agents en limitant le nombre d'alternatives lexicales rencontrées. Cette étude souligne l'importance de considérer les contraintes du monde réel lors de l'application de modèles théoriques à des systèmes robotiques.

### D. Évolution Culturelle et Adaptation Comportementale

Au-delà de la simple émergence d'un lexique, les principes du NG s'inscrivent dans un cadre plus large d'évolution culturelle, où des conventions ou des comportements se propagent et se fixent au sein d'une population. Steels (1995) [3] concevait déjà le langage comme un système adaptatif autonome, façonné par des interactions localisées. Des travaux plus récents ont explicitement utilisé les mécanismes inspirés du NG pour l'adaptation comportementale dans les essaims de robots.

Cambier et al. (2020) [5] proposent une perspective sur l'évolution du langage en robotique en essaim, arguant que l'auto-organisation culturelle, par opposition à l'évolution biologique, est un mécanisme clé pour développer des compétences de communication complexes. Ils discutent de divers jeux de langage, dont le NG, comme outils pour atteindre cet objectif.

Dans une étude empirique, Cambier et al. (2021) [6] ont démontré comment les principes de l'évolution culturelle, s'appuyant sur une variante du MNG (hearer-only), pouvaient être utilisés pour l'adaptation des paramètres d'un comportement d'agrégation probabiliste chez des Kilobots. Ici, les "mèmes" échangés ne sont pas des noms d'objets, mais des paramètres comportementaux. Les interactions locales et la propagation culturelle permettent à l'essaim d'adapter collectivement son comportement aux conditions macroscopiques, illustrant la puissance des jeux de langage pour l'évolution de comportements complexes.

De même, Belorgey et al. (2021) [7] ont exploré l'adaptation de comportements pour des essaims de robots autonomes en introduisant des biais (succès, nouveauté, mutation) dans les jeux de langage, s'inspirant des travaux en anthropologie et psychologie évolutionniste. Leur modèle, appliqué à une tâche de recherche de cibles, montre comment un groupe d'agents peut apprendre et réapprendre des comportements adaptés à un environnement stationnaire ou non-stationnaire.

### E. Positionnement du Projet

Le présent projet s'appuie sur ces avancées significatives. En se concentrant sur l'implémentation du Naming Game sur des robots Kilobot, il vise à explorer les dynamiques d'émergence du consensus dans un contexte physique réel. Il s'agira notamment d'étudier le naming game dans un cadre simplifié (objets limités et mots représentés par des nombres entre 0 et 255).

## III. Méthodologie

Dans le cadre de ce projet, nous nous sommes intéressés à l'émergence de consensus au sein d'essaims de robots. Pour cela, nous avons travaillé d'une part avec des Kilobots, et d'autre part avec l'algorithme du Naming Game.

Le choix des Kilobots s'explique par leur simplicité : ce sont de petits robots rudimentaires, bien adaptés à l'étude de systèmes collectifs simples, avec la possibilité d'étendre ensuite ces travaux à des systèmes plus complexes. Quant au Naming Game, il s'agit d'un modèle particulièrement pertinent pour mettre en évidence les mécanismes menant à l'apparition d'un consensus au sein d'un groupe.

Notre travail s'est principalement appuyé sur deux supports : les simulations sur ordinateur à l'aide du simulateur Kilombo [10], et des expérimentations réelles avec les robots eux-mêmes. Enfin, pour faire le lien entre ces deux approches, nous avons utilisé l'analyse d'images afin de détecter la position des Kilobots et de suivre l'émergence du consensus.

Dans la suite de cette partie, nous exposons les divers méthodes que nous avons employé afin de démontrer l'émergence de consensus au sein d'un essaim de robots.

### A. Kilobots

Dans cette partie, nous présentons le travail réalisé avec les Kilobots. Celui-ci se divise en trois axes principaux : la prise en main du matériel et de l'environnement de travail, la programmation des Kilobots, et enfin la simulation à l'aide du simulateur Kilombo.

*1) Matériel:* Afin de mener à bien notre projet, nous avons eu accès au matériel suivant :

1) **Une trentaine de Kilobots**
   Les Kilobots sont les principaux outils utilisés dans ce projet. Il s'agit de petits robots autonomes composés d'une batterie, de moteurs permettant le déplacement par vibration, d'une LED, ainsi que d'un émetteur et récepteur infrarouge (IR). Grâce à leur dispositif IR, ils sont capables de communiquer avec d'autres Kilobots dans un rayon d'environ 7cm.
2) **Un OHC (*Overhead Controller*)**
   Grâce à sa LED infrarouge, l'OHC (Figure 3) permet de transmettre aux Kilobots les programmes à exécuter. Il joue un rôle central dans le déploiement des algorithmes sur les robots.
3) **Une arène**
   L'arène est constituée de l'OHC, de barrières délimitant l'espace d'évolution des robots, ainsi que d'une caméra, utilisée pour l'analyse d'images et la détection des Kilobots.
4) **Un poste de travail**
   Ce poste comprend un ordinateur et un second écran, facilitant le développement, les tests et les ajustements des programmes en temps réel. C'est également via ce poste, à l'aide de l'interface Kilogui et de l'OHC, que les programmes sont transmis aux Kilobots.

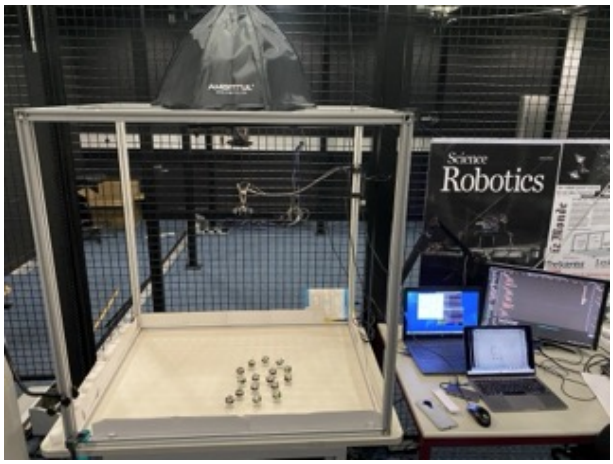


Fig. 2. Arène et poste de travail.

Avant de débuter véritablement le projet, nous avons dû nous familiariser avec l'ensemble de ce matériel, et plus particulièrement avec les Kilobots et l'OHC, afin de pouvoir les manipuler efficacement et leur faire exécuter les programmes que nous avons développés.

Fig. 3. Overhead Controller.

*2) Programmation des Kilobots et simulation avec Kilombo:* La programmation des Kilobots a été abordée de manière **incrémentale**, en commençant par des comportements simples, testés d'abord dans un environnement simulé, avant de passer à l'expérimentation réelle.

*a) Utilisation du simulateur Kilombo:* Pour tester nos programmes, nous avons utilisé *Kilombo*, un simulateur dédié aux Kilobots. Ce simulateur permet d'observer le comportement de plusieurs robots virtuels, facilitant ainsi le débogage et la mise au point des algorithmes, sans les contraintes matérielles liées aux robots physiques.

Notre premier programme consistait simplement à faire clignoter la LED d'un Kilobot. Nos recherches pour cet exercice introductif nous a permis de mettre en évidence des **différences importantes entre l'environnement simulé et réel**. En particulier :

- La fonction `delay()` ne fonctionne pas dans le simulateur.
- La fonction `set_motors(uint8_t left, uint8_t right)` fonctionne de manière très limitée. Dans Kilombo, les valeurs de l'un des paramètres doit impérativement appartenir à l'intervalle `[68, 90]` pour produire un mouvement visible du robot, alors que sur les robots réels, cette fonction fonctionne comme attendu sur une plage plus large.

Malgré ces limitations, Kilombo s'est révélé **très utile pour le développement**, car il permet d'observer simultanément plusieurs robots et de diagnostiquer plus facilement les comportements inattendus, ce qui serait plus fastidieux avec un seul robot physique connecté au poste de travail.

*b) Progression de la programmation:* Une fois la gestion de la LED maîtrisée, nous avons poursuivi avec la **programmation du mouvement** des Kilobots. L'objectif était de tester différents déplacements et rotations, toujours dans une logique de complexification progressive.

La dernière étape, et la plus critique pour notre projet, fut la mise en œuvre de la **communication entre robots**. Cette phase a également été abordée de façon incrémentale : en commençant par l'envoi de messages simples, puis en implémentant la réception, avant de gérer les deux simultanément.

La communication entre Kilobots repose sur l'utilisation de **fonctions de rappel** (*callbacks*) que l'on enregistre pour réagir à certains événements. Les principales fonctions sont :

- `message_tx()`, appelée pour transmettre un message;
- `message_tx_success()`, déclenchée lorsque l'envoi d'un message a été effectué avec succès;
- `message_rx()`, appelée lorsqu'un message est reçu.

La maîtrise de ces fonctions est essentielle, car elles forment la base des **interactions entre robots**, indispensables à l'émergence de comportements collectifs, tels que le *consensus*.

*c) Vers des comportements collectifs:* Une fois toutes ces étapes validées — gestion des LED, mouvements, communication — aussi bien sur simulateur que sur les robots réels, nous avons pu entamer la phase centrale de notre projet : la **programmation du *Naming Game***, un algorithme d'émergence de consensus, qui fera l'objet d'une analyse détaillée dans la suite de ce rapport.

### *B. Analyse d'image*

En complément des fonctionnalités précédemment décrites, nous avons intégré une composante d'analyse d'image permettant de localiser les robots dans l'arène. Cela est nécessaire pour ajouter des éléments de réalité augmentée, car les robots eux-mêmes ne fournissent pas de données en retour. L'analyse d'image est également utilisée pour détecter le moment où un consensus est atteint.

Nous avons développé un serveur dédié à la gestion de la caméra, qui filme l'ensemble de l'arène. Ce serveur permet plusieurs actions : démarrer la caméra via la route `/start`, l'arrêter via `/stop`, capturer une image via `/capture_image`, modifier la résolution via `/set_resolution/<mode>`, et sélectionner un autre périphérique de capture via `/set_camera/int:new_camera_id`.

Cependant, une image statique s'est rapidement révélée insuffisante pour répondre aux exigences de notre cas d'usage. Nous avons donc fait évoluer le système afin de permettre une capture d'image en temps réel, couplée à un module de détection de cercles qui sera davantage expliqué par la suite.

Pour garantir la précision des mesures extraites, un calibrage (Figure 8) rigoureux a été nécessaire. Celui-ci permet d'estimer une échelle en pixels par centimètre à partir de la superposition d'une grille physique, placée sous la caméra, avec une grille virtuelle affichée à l'écran. L'alignement des deux grilles est facilité par un curseur permettant d'ajuster la taille de la grille virtuelle.

Plus précisément, la grille physique utilisée pour le calibrage est constituée de cercles régulièrement espacés. Pour amorcer le processus, nous avons identifié les cercles situés aux extrémités supérieure et inférieure de la grille, puis tracé la droite les reliant, correspondant à l'une des deux diagonales principales de la structure en carton. En connaissant les caractéristiques géométriques exactes de cette grille — notamment le nombre de cercles présents sur la diagonale, leur espacement et leur rayon — il devient possible de reconstruire virtuellement l'ensemble des cercles de cette diagonale. Cette reconstruction est réalisée à partir des seules coordonnées des deux cercles extrêmes identifiés, ce qui permet un positionnement précis de la grille virtuelle sur l'image capturée.

L'ajustement de l'échelle consiste alors à modifier dynamiquement les paramètres de la grille virtuelle - en particulier le facteur d'échelle et le rayon des cercles - afin d'aligner au mieux les cercles affichés à l'écran avec leurs

homologues physiques visibles sur l'image capturée. Le but est d'obtenir une superposition aussi précise que possible entre les cercles réels et virtuels de la diagonale de référence, ce qui garantit que l'échelle calculée soit fiable. Cette échelle est ensuite utilisée pour convertir les coordonnées et dimensions détectées (telles que les centres et rayons des cercles correspondant aux Kilobots) en unités physique exploitables.

La détection des cercles repose sur la transformée de Hough circulaire, une méthode standard en traitement d'image pour l'identification de formes géométriques. Cette approche reformule la détection d'un cercle comme une recherche dans un espace de paramètres tridimensionnel $(x, y, r)$. Chaque pixel détecté comme contour (via l'algorithme de Canny, détaillé plus bas) est utilisé pour émettre des hypothèses sur la présence de cercles. L'algorithme calcule les centres possibles des cercles passant par chaque point bordé, et incrémente les cellules correspondantes dans l'espace accumulateur. Les pics locaux de cet espace correspondent aux cercles ayant reçu le plus de votes, c'est-à-dire ceux dont le contour est le plus vraisemblablement présent dans l'image.

L'algorithme de Canny est utilisé pour la détection des contours, une méthode robuste en vision par ordinateur. Il consiste en plusieurs étapes : floutage de l'image pour réduire le bruit, calcul du gradient d'intensité pour détecter les bords, suppression des non-maxima pour affiner les contours à une largeur d'un pixel, et hystérésis avec deux seuils : un seuil haut pour valider les contours forts et un seuil bas pour prolonger ceux qui leur sont connectés. Cette méthode permet de minimiser les faux positifs tout en préservant les bords réels.

La détection (Figure 7) peut être activée par une simple pression sur la touche "D". Cette étape a été soigneusement adaptée à notre environnement de test (configuration de l'arène, positionnement de la caméra, conditions d'éclairage), et repose sur la fonction `HoughCircles` de la bibliothèque OpenCV, qui implémente la transformée de Hough circulaire.

Cinq paramètres essentiels ont été ajustés pour obtenir une détection à la fois fiable et robuste. Les paramètres `min_rad` et `max_rad` définissent respectivement les rayons minimaux et maximaux des cercles détectables, ce qui permet de cibler spécifiquement les Kilobots. Le paramètre `dmax` impose une distance minimale entre deux cercles détectés afin de limiter les doublons. Enfin, `hough_param_1` contrôle le seuil supérieur utilisé lors de la détection des contours via l'algorithme de Canny, tandis que `hough_param_2` fixe le seuil dans l'espace accumulateur : seuls les cercles ayant accumulé un nombre suffisant de votes y sont validés. Ce mécanisme à double seuil offre un compromis efficace entre sensibilité de détection et limitation des fausses détections dues à des facteurs extérieurs.

L'ensemble de ces paramètres est enregistré dans un fichier de configuration, mais reste modifiable dynamiquement via des curseurs, permettant d'observer en temps réel l'impact de chaque ajustement sur le résultat de la détection.

L'obtention d'une échelle fiable constitue l'étape la plus délicate. Une fois celle-ci correctement déterminée, nous avons optimisé le rendu visuel de la détection afin de le rendre aussi informatif que possible. Le système permet désormais l'affichage des cercles de communication, la détection des LEDs, l'identification individuelle des Kilobots (liée à leur position), la visualisation des liens de communication entre robots, l'émission d'un message lorsque toutes les LEDs convergent vers une même couleur, ainsi que l'ajout d'une grille facilitant la localisation précise des robots dans l'arène.

### C. Naming Game

*1) Univers:* Pour notre implémentation du Naming Game avec les Kilobots, nous avons décidé d'utiliser un univers plus simple avec un seul objet à nommer. Cela permet de mieux visualiser et comprendre les dynamiques menant à un consensus dans le groupe de robots. En effet, réduire le nombre d'objets à nommer limite la complexité des interactions puisqu'il n'est pas nécessaire de choisir le mot entre les plusieurs objets, mais uniquement de se référer à l'objet unique présent dans l'environnement. De plus, ce choix favorise une convergence plus rapide vers un nom commun, facilitant ainsi l'analyse du comportement collectif. Ainsi l'analyse des mécanismes fondamentaux du jeu : invention de noms, transmission, adoption ou rejet selon le succès des interactions, en sera facilitée. L'objectif étant d'arriver à un consensus sur le mot associé à l'objet commun, chaque Kilobot doit commencer par sélectionner aléatoirement un mot parmi une liste fournie à tous les robots.
Étant donné la simplicité des Kilobots, équipés d'un micro-contrôleur ATmega 8 bits, nous avons choisi de symboliser l'objet à nommer et les mots par des entiers codés sur 8 bits, pour éviter de surcharger les Kilobots.

*2) Couleurs:* Avoir les mots associés à des entiers permet également d'associer une couleur à chaque mot. Cela permet de suivre plus facilement l'évolution vers le consensus de manière visuelle.
La LED de chaque Kilobot est en RGB et peut prendre jusqu'à 4 valeurs pour chaque couleur soit 64 couleurs possibles. Pourtant, nous avons décidé d'utiliser une liste de seulement 10 couleurs pour pouvoir différencier les couleurs les unes des autres. L'association mot/couleur est fixée par une liste des couleurs en RGB dont les indices correspondent aux différents mots. Ainsi le Kilobot peut fixer sa LED à la couleur du mot qu'il a choisi, que ce soit sur simulateur ou sur les Kilobots (Figure 4). On peut donc voir lorsqu'un consensus est atteint puisque tous les Kilobots prennent la même couleur.

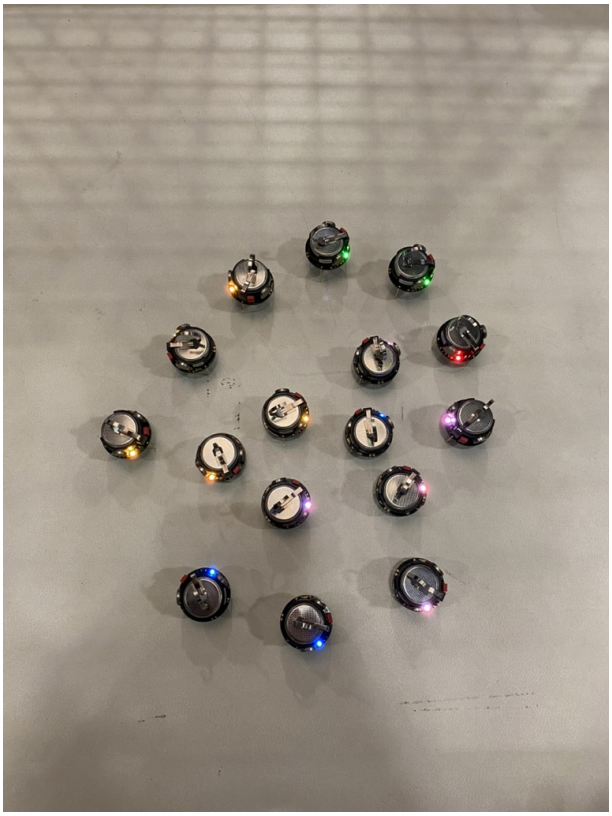

Fig. 4. Kilobots affichant la couleur correspondant à leur mot choisi.

*3) Rôles:* Les robots doivent échanger leurs mot associé à l'objet, ils doivent donc tous pouvoir recevoir le mot choisit par des Kilobots dans leur rayon de communication et également leur partager le sien. Il faut alors qu'il changent entre les 2 états possibles : auditeur (listener) et orateurs (speaker). Toutes les 4 secondes, chaque Kilobot réévalue sont état : il choisit aléatoirement si il change d'état avec une probabilité de 20% ou si il le conserve (80%).

*4) Échanges:* Le déroulé du jeu repose sur des messages transmis par les robots en état d'orateurs (speaker) aux robots auditeurs (listener) proches. Chaque message contient un identifiant de 8 bits représentant un mot. Lorsqu'un robot reçoit un message, il compare le mot reçu avec celui présent dans sa mémoire. Il génère ensuite un nombre aléatoire entre 0 et 9, si le nombre généré est strictement supérieur à 6, le Kilobot choisit le mot qui vient de lui être transmis avec une probabilité, sinon il conserve son mot. Cette probabilité de 30% a été choisie empiriquement pour assurer une diffusion suffisante des mots tout en permettant aux robots de conserver une certaine "mémoire", il s'agit d'un compromis entre diversité et convergence vers un consensus.
À chaque réception ou envoi d'un message, on bloque la communication du robot pour avoir le temps de le traiter, éviter d'avoir trop de messages envoyés ou reçus en même temps par un Kilobot.

*5) Pseudo code:* Ci-dessous le pseudo-code de l'implémentation du Naming Game :

**Algorithm 1** Algorithme du Naming Game adapté

```
procedure INITIALISER
    état ← LISTENER
    objet ← OBJET_CIBLE
    mot_actuel ← mot_aléatoire()
    dernier_message ← NULL
    dernier_changement ← temps_actuel()
procedure NAMINGGAMELOOP
    while non CONSENSUSATTEINT do
        if état = SPEAKER then
            if TEMPS_DEPUIS(dernière_interaction) ≥ DÉLAI_DIFFUSION then
                ENVOYER_MESSAGE(objet, mot_actuel)
                dernière_interaction ← temps_actuel()
        else if état = LISTENER then
            if MESSAGE_REÇU then
                (objet_r, mot_r) ← LIRE_MESSAGE
                if mot_r ≠ mot_actuel and PROBA(0.4) then
                    mot_actuel ← mot_r
        if TEMPS_DEPUIS(dernier_changement) ≥ DÉLAI_CHANGEMENT_ETAT and PROBA(0.2) then
            état ← CHANGER_ÉTAT_ALÉATOIREMENT
            dernier_changement ← temps_actuel()
function CONSENSUSATTEINT
    return TOUS_LES_ROBOTS_ONT_MÊME_MOT
```

*6) Version déplacements:* Nous avons également implémenté une version où les Kilobots se déplacent.
Si un Kilobot qui est dans un état d'auditeur n'a pas reçu de message depuis 4 secondes, il amorce un déplacement dans une direction aléatoire, dans le but de se rapprocher potentiellement d'autres robots avec lesquels il pourrait interagir. Au bout de 2 secondes à se déplacer, il s'arrête et repasse en mode écoute.

*7) Mesures et tests:* Pour réaliser nos tests, nous avons utilisé les différents types de topologies pour en mesurer le temps de convergence.

Notre problématique étant d'évaluer combien de temps il faut à un groupe de N robots pour parvenir à un consensus, nous avons voulu comparer nos résultats en fonction du nombre de Kilobots étant impliqués dans le jeu et de la topologie de départ. Nous avons donc étudié 5 configurations : en rond, ligne, tas, croix et en clusters reliés avec entre 2 et 30 Kilobots. Nous avons choisi de garder le même univers pour tous les tests : 10 mots possibles pour 1 objet à nommer.

Faire varier le nombre de robot n'est pas toujours applicable puisque la densité doit rester la même. C'est pourquoi dans les clusters reliés ou dans la configuration en croix,

l'augmentation du nombre de Kilobots est incrémenté du nombre nécessaire pour garantir la stabilité de la densité de robots dans les clusters ou branches de croix. Ainsi on passe d'une croix à 5 Kilobots à une croix (Figure 5) à 9 Kilobots (Figure 6) pour ajouter un Kilobot de chaque côté de la croix.

Fig. 5. Configuration en croix à 5 Kilobots.

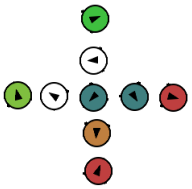

Fig. 6. Configuration en croix à 9 Kilobots.

## IV. Résultats

### A. Analyse d'images

Les expérimentations menées ont permis de valider la mise en place fonctionnelle du serveur de gestion de la caméra. L'ensemble des points d'accès définis — démarrage, arrêt, capture d'image, modification de la résolution et sélection du périphérique vidéo — ont été testés et se sont avérés pleinement opérationnels. Le système assure ainsi un contrôle complet et dynamique du flux vidéo.

La phase de calibrage, bien qu'exigeante, s'est révélée déterminante pour garantir la précision des mesures. Une fois ce calibrage effectué avec soin, la détection des cercles à l'aide de la transformée de Hough s'est montrée fiable et suffisamment précise pour répondre aux exigences du projet. La (Figure 7) illustre les performances de l'algorithme de détection dans un environnement expérimental représentatif.

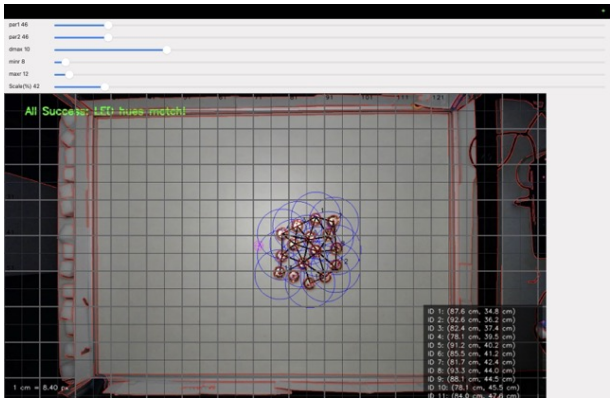


Fig. 7. Détection de cercle.

Le système offre une visualisation en temps réel des cercles détectés, assortie d'une interface interactive permettant d'ajuster dynamiquement les paramètres critiques. Cette flexibilité a été particulièrement utile pour adapter la détection aux contraintes spécifiques de notre arène, notamment en termes de luminosité, d'angle de prise de vue et de positionnement des robots.

Enfin, les fonctionnalités de visualisation avancées — telles que l'affichage des LEDs, l'identification des Kilobots, la représentation des liens de communication ou encore la détection automatique d'un consensus lumineux — ont toutes été intégrées avec succès. Ces éléments contribuent à une analyse visuelle claire et informative, essentielle pour le suivi en temps réel des comportements collectifs dans l'arène.

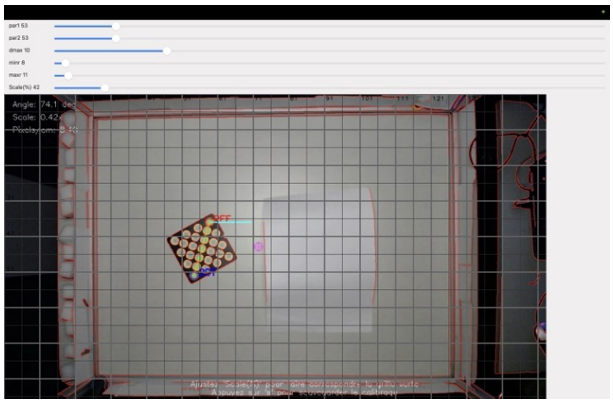

Fig. 8. Calibrage de la caméra.

### B. Naming Game

*1) Résultats obtenus avec le Naming Game:* Notre objectif était d'évaluer le temps nécessaire à un groupe de $N$ Kilobots pour parvenir à un consensus. Pour cela, nous avons comparé les performances du *Naming Game* en fonction du **nombre de robots** impliqués et de leur **configuration initiale dans l'espace**.

Nous avons étudié cinq topologies différentes : *rond*, *ligne*, *tas*, *croix* et *clusters* connectés, pour des groupes allant de 2

à 30 Kilobots.
Les résultats que nous avons obtenus sont présentés sur la figure (Figure 9).

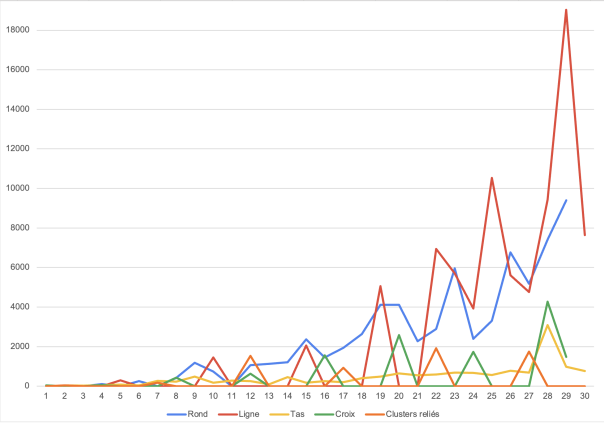


Fig. 9. Temps pour atteindre le consensus en fonction du nombre de Kilobots et de la configuration.

On retrouve les moyennes de temps nécessaire à l'obtention d'un consensus pour différents nombre de robots. En bleu on retrouve la configuration en rond, en rouge la ligne, en jaune le tas, en vert la croix et en rouge les clusters reliés.

*a) Impact de la topologie sur la convergence:* Les résultats ont montré que la **topologie initiale a une influence significative** sur la rapidité d'émergence du consensus. Plus la structure de départ se rapproche d'un *graphe complet* — c'est-à-dire où chaque robot peut facilement interagir avec un grand nombre de voisins — plus le consensus est atteint rapidement. C'est par exemple le cas avec la configuration en **tas**, où les temps de convergence sont parmi les plus faibles.

À l'inverse, lorsque la topologie offre peu de connexions entre les robots, comme dans la configuration en **ligne**, le temps nécessaire pour atteindre un consensus global augmente considérablement. Cela s'explique par la *faible densité de communication*, qui ralentit la propagation des informations entre les membres du groupe.

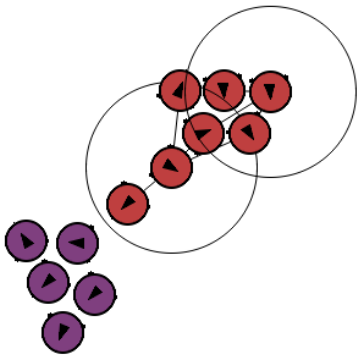

Fig. 10. Émergence d'un consensus local au sein d'une configuration à 2 clusters reliés.

*b) Consensus locaux dans les clusters:* Une observation intéressante a été faite dans la configuration en **clusters reliés**. Dans cette topologie, les Kilobots ont tendance à former rapidement des **consensus locaux** au sein de chaque sous-groupe (ou cluster) (Figure 10). Ces consensus partiels sont généralement atteints rapidement et de manière stable. En revanche, le passage à un consensus global, impliquant l'ensemble des robots, prend beaucoup plus de temps. Cela souligne le rôle des *liaisons inter-clusters* dans la diffusion de l'information à l'échelle de tout l'essaim.

*c) Comparaison simulation / robots réels:* Nous avons également testé notre implémentation du *Naming Game* sur les Kilobots réels. Les résultats obtenus étaient globalement cohérents avec ceux observés en simulation, bien que certaines dynamiques soient plus difficiles à percevoir dans un environnement physique.

La simulation via `Kilombo` offre certains avantages, notamment la possibilité **d'accélérer le temps**, ce qui facilite l'analyse de comportements émergents à long terme. Ce n'est évidemment pas possible avec les robots physiques.

Par ailleurs, nous avons été confrontés au phénomène bien connu du **reality gap**, qui désigne l'écart entre les comportements attendus en simulation et ceux observés en conditions réelles. Par exemple, dans le simulateur, les mouvements des Kilobots sont parfaitement linéaires et conformes au modèle théorique. En pratique, de nombreux facteurs influencent leurs déplacements : la courbure de la table, l'usure ou la forme des pieds des robots, ou encore des variations dans la friction. Ces éléments rendent les comportements réels moins prévisibles et parfois moins efficaces, ce qui peut impacter le temps de convergence du consensus.

## V. Conclusion

Ce travail a permis d'étudier la formation d'un consensus linguistique dans un essaim de robots Kilobots à travers l'implémentation du protocole du Naming Game. L'approche expérimentale, fondée sur une programmation embarquée adaptée aux contraintes matérielles des robots, l'usage du simulateur Kilombo et un système de suivi visuel en temps réel, a permis d'observer et de mesurer la dynamique de convergence dans différentes configurations.

Les résultats montrent que la topologie du réseau d'interaction influence fortement le temps nécessaire pour atteindre un consensus. Les formations densément connectées, comme les amas, permettent une propagation rapide des conventions, tandis que les configurations plus dispersées, telles que les lignes ou les clusters peu reliés, ralentissent la convergence. Ce ralentissement est souvent dû à l'apparition de consensus locaux dans certaines zones de l'essaim, suivie d'une phase plus lente de diffusion globale. Ces observations sont cohérentes avec les résultats attendus du modèle théorique et confirment, dans un contexte incarné, l'importance de la structure des interactions pour les dynamiques collectives.

L'implémentation a nécessité des simplifications, notamment une représentation des mots par des entiers sur 8 bits et un unique objet à nommer. Ces choix ont permis de maîtriser la complexité du système et de se concentrer sur les mécanismes fondamentaux du consensus. Néanmoins, ils limitent l'applicabilité du protocole à des contextes plus riches ou variés. Par ailleurs, des écarts ont été observés entre les résultats obtenus en simulation et ceux mesurés sur les robots physiques. Ces différences s'expliquent en grande partie par les imprécisions du déplacement, les variations dans la détection des messages, et les effets de l'environnement réel.

Plusieurs prolongements sont envisageables. La modification du rayon de communication permettrait d'étudier comment la portée des interactions influence la rapidité et la robustesse du consensus, et d'évaluer les seuils de connectivité minimaux nécessaires à son émergence. L'introduction d'éléments de réalité augmentée, tels que des obstacles projetés dans l'arène, offrirait un moyen de tester le comportement des robots dans des environnements plus dynamiques, en introduisant des contraintes d'évitement ou des conventions contextuelles. Enfin, l'extension du protocole à plusieurs objets ou à des structures linguistiques plus complexes constituerait une étape vers l'étude de formes de communication collective plus élaborées.

Ce travail fournit ainsi un cadre expérimental pour analyser les conditions d'émergence d'un consensus dans un essaim physique, tout en soulignant les limites actuelles du système et les leviers possibles pour les dépasser.

## Remerciements

Nous souhaitons remercier M. Boumaza, notre référent, qui nous a permis d'effectuer ce projet et nous a aidé et conseillé sur les différentes directions à prendre pendant le déroulement du projet.

Nous remercions également Chloé B., Damien S., Divya P., pour leur relecture éclairée.

## References

[1] Baronchelli, A., Loreto, V., & Steels, L. (2008). **IN-DEPTH ANALYSIS OF THE NAMING GAME DYNAMICS : THE HOMOGENEOUS MIXING CASE**. International Journal Of Modern Physics C, 19(05), 785-812. https://doi.org/10.1142/s0129183108012522

[2] Trianni V., De Simone D., Reina A., & Baronchelli A. (2016). **Emergence of Consensus in a Multi-Robot Network : From Abstract Models to Empirical Validation**. IEEE Robotics And Automation Letters, 1(1), 348-353. https://doi.org/10.1109/lra.2016.2519537

[3] Steels, L. (1995). **A Self-Organizing Spatial Vocabulary.** Artificial Life, 2(3), 319-332. https://doi.org/10.1162/artl.1995.2.319

[4] Baronchelli, A. (2011). **The minimal Naming Game: a complex systems approach.** book chapter, ed. Luc Steels. (due out, 2012).

[5] Cambier, N., Miletitch, R., Frémont, V., Dorigo, M., Ferrante, E., Trianni, V. (2020).**Language Evolution in Swarm Robotics: A Perspective.** Frontiers In Robotics And AI, 7. https://doi.org/10.3389/frobt.2020.00012

[6] Cambier, N., Albani, D., Frémont, V., Trianni, V., Ferrante, E. (2021). **Cultural evolution of probabilistic aggregation in synthetic swarms.**Applied Soft Computing, 113, 108010. https://doi.org/10.1016/j.asoc.2021.108010

[7] Belorgey, R., Aquilanti, L., Plantec, E. (2021). **Vers l'adaptation de comportements par le biais de l'évolution culturelle pour des essaims de robots autonomes.** Projet tutoré, Master 1 Sciences Cognitives

[8] Rubenstein, M., Ahler, C., Hoff, N., Cabrera, A., Nagpal, R. (2014). **Kilobot: A low cost robot with scalable operations designed for collective behaviors.**Robotics And Autonomous Systems, 62(7), 966-975. https://doi.org/10.1016/j.robot.2013.08.006

[9] Klügl, F. (2001). **Swarm Intelligence : From Natural to Artificial Systems by Eric Bonabeau, Marco Dorigo and Guy Theraulaz .** Journal Of Artificial Societies And Social Simulation, 4. http://jasss.soc.surrey.ac.uk/4/1/reviews/kluegl.html

[10] Jansson, F., Hartley, M., Hinsch, M., Slavkov, I., Carranza, N., Olsson, T. S., ... Grieneisen, V. A. (2015). **Kilombo: a Kilobot simulator to enable effective research in swarm robotics.** arXiv preprint arXiv:1511.04285.

## VI. ANNEXES

### *A. Annexe 1 - Algorithme canonique du Naming Game*

Ci-dessous le pseudo code général de l'implémentation du Naming Game :

**Algorithm 2** Algorithme canonique du Naming Game en pseudo code

```
1:  procedure NAMINGGAME
2:      // Initialisation
3:      for all agent ∈ agents do
4:          agent.inventory ← ∅

5:      while not globalConsensus do
6:          speaker ← selectRandomAgent()
7:          listener ← selectRandomAgent()
8:          if speaker.inventory = ∅ then
9:              proposedName ← inventNewName()
10:         else
11:             proposedName ← selectRandom-
    Name(speaker.inventory)
12:         if proposedName ∈ listener.inventory then
13:             listener.inventory ← {proposedName}
14:             speaker.inventory ← {proposedName}
15:         else
16:             listener.inventory ← listener.inventory
    ∪{proposedName}

17:         globalConsensus ← true
18:         firstInventory ← agents[0].inventory
19:         for all agent ∈ agents do
20:             if agent.inventory ≠ firstInventory then
21:                 globalConsensus ← false
22:                 break
```