Recherche décentralisée de connexité pour réseaux de capteurs mobiles

Merwan Achibet

Introduction

On imagine le problème suivant : un groupe de n capteurs mobiles est réparti aléatoirement dans un espace aérien. On part de l'hypothèse qu'un capteur connaît uniquement le nombre total de capteurs du système et qu'il est assez sophistiqué pour pouvoir déterminer précisément sa position absolue. Les capteurs sont dotés de matériel de communication sans fil et peuvent s'envoyer des messages à condition que la distance les séparant soit inférieure à leur rayon d'émission R_c.

Le réseau constitué par cet essaim d'appareils volants forme un graphe dynamique dont les nœuds sont les capteurs. Deux nœuds sont reliés par un arc si les capteurs qui leur sont associés sont en mesure de communiquer, c'est à dire s'ils sont assez proches. La figure 1 décrit un exemple de scénario impliquant trois capteurs.

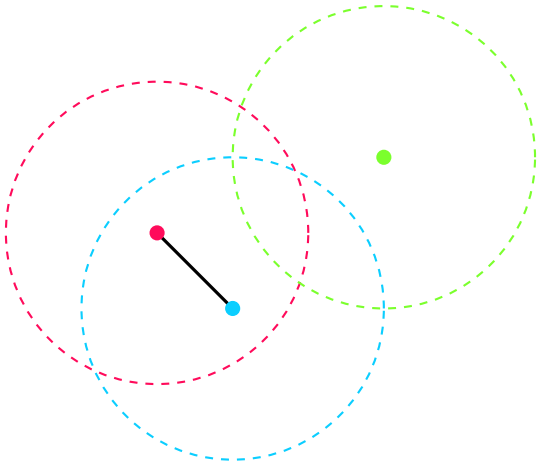


FIGURE 1 – Les capteurs rose et bleu peuvent communiquer et sont connectés tandis que le capteur vert est isolé.

On considère un capteur comme un agent autonome capable de se mouvoir dans l'espace. Afin de ne pas se préoccuper de considérations géométriques superflues, il est supposé qu'un capteur conserve toujours la même altitude et donc on limite son déplacement à deux dimensions. La contrainte principale de cet exercice est que l'on refuse toute forme de contrôle global sur l'essaim de capteurs. Toutes les actions entreprises par un appareil seront uniquement dû à ses décisions propres et dépendront de la vue réduite du système dont il dispose.

La possibilité pour un capteur de communiquer avec ses semblables est au centre de nos préoccupations car on considère qu'un capteur isolé est inutile puiqu'incapable de partager des données. Dans le contexte de l'étude décentralisée d'un graphe dynamique, deux questions se posent :

1. Comment déterminer si le réseau de capteurs est connexe ?
2. Comment déplacer les capteurs de façon à ce qu'il le devienne ?

Pour répondre à la première question, on se concentre sur les communications de capteur à capteur en proposant une méthode pour laquelle chaque agent diffuse sa vision de la connexité du réseau, étiquettée en fonction de l'origine de l'information et de sa date. Un mécanisme de filtrage autorégulant sera mis en place pour répandre les informations critiques telles que l'ajout d'un nouveau nœud à une composante connexe.

La réponse à la seconde question s'attache à l'aspect mobile d'un capteur. On propose une méthode de guidage décentralisé se basant sur l'imitation de plusieurs lois physiques de la nature afin de former un maillage à la fois connexe, équilibré et étalé dans l'espace. Finalement, on envisage une extension de ce système pour permettre aux capteurs d'éviter

naturellement les obstacles.

1 Déterminer si le réseau est connexe

Dans cette section, on se place du point de vue d'un capteur c et l'on considère que le réseau est connexe lorsque c le pense connexe.

Les capteurs étant mobiles, le réseau de communication qu'ils forment est hautement dynamique et des événements impromptus, comme le déplacement d'un capteur ou une panne, peuvent à tout moment faire varier la connexité du graphe résultant. Comme énoncé précédemment, c a connaissance du nombre de capteurs n du système. Déterminer la connexité du réseau revient donc à un problème de comptage décentralisé, où $\tilde{n}$ est le nombre de capteurs que c sait connectés à sa propre composante connexe, et le graphe est connexe si $\tilde{n} = n$.

L'idée que l'on présente ici repose sur un partage permanent, entre voisins, de la vision du réseau que chaque capteur entretient. Si l'on pouvait traduire en langue humaine un message de c à c', on entendrait :

> *Je suis c et je sais que x à été vu dans le réseau il y a 14 secondes. Par contre, je n'ai pas eu de nouvelles de mon voisin y depuis un certain temps, je doute qu'il soit toujours présent dans le réseau.*

Dans notre méthode, un capteur transmet régulièrement à ses voisins son avis sur la présence ou non d'un nœud dans leur composante connexe. Afin de permettre de gérer des informations contradictoires, une date et dans certains cas, une source, sont associées à chaque information de nœud. L'objectif envisagé est de concevoir un système dans lequel les données sont diffusées par transmission entre voisins, fusion des informations de présence et filtrage des informations contradictoires.

Un message adopte toujours la même structure

On présente d'abord les trois structure de données qui peupleront la mémoire à chaque capteur.

Liste de voisins directs

La première liste, N, représente le voisinage de c et contient uniquement les identifiants des nœuds en liaison directe avec ce dernier.

Un nœud est ajouté à N si c en reçoit un message et qu'il n'était pas déja présent dans la liste. Un nœud est retiré de N si c n'en a pas reçu de message depuis un certain temps, c'est le mécanisme de surveillance.

On choisit ici de rendre les capteurs proactifs : leurs messages sont envoyés volontairement à intervalle régulier afin d'informer leurs voisins de leur présence. Ainsi, on considère qu'un capteur a potentiellement disparu s'il n'envoie plus de message.

Liste de nœuds connectés

La seconde liste entretenue par un capteur est C. Elle représente sa vision de la connexité du réseau, soit concrètement, tous les nœuds qui, de son point de vue, y sont connectés directement (par la liaison sans-fil) ou indirectement (par l'intermédiaire d'autres capteurs). On peut donc en déduire que tous les nœuds de N apparaissent dans C.

Cependant, et contrairement à N, C ne contient pas que des identifiants. Ses entrées sont de la forme (x, t, s) avec x l'identifiant du nœud dont l'on veut exprimer la présence sur la composante connexe, t le dernier temps auquel x a été détecté et s le nœud à l'origine de cette information.

Par exemple, si un capteur a dans sa liste C l'entrée $(y, 52, z)$, cela signifie qu'il sait que le capteur y fait partie de la même composante connexe que lui car il a été détecté au temps 52 et l'information a été relayée par z. Il est important de différencier z, le nœud qui a transmis l'information au capteur courant (un voisin donc) du nœud distant qui a détecté y au temps 52 et dont l'identité ne nous importe pas.

Liste de doutes

La troisième et dernière liste, D, contient une liste de capteurs dont c n'est plus certain de la connexité. Ses entrées prennent la forme de paires (x, t) où x est le nœud sur lequel le doute se pose et t la date à laquelle l'absence de x a été dernièrement remarquée.

On ajoute une entrée (c', t) à D lorsque c' est présent dans la liste N des voisins mais que, par

surveillance, on se rend compte au temps t qu'il n'a pas communiqué depuis un certain temps. À l'inverse, cette même paire migre de la liste de doutes vers la liste de connexité si une nouvelle communication est plus tard détectée (le t est alors mis à jour pour réfléter ce changement).

Il est évident qu'un nœud ne peut être présent à la fois dans C et D. Ce constat nous permet d'évaluer l'espace mémoire nécessaire à $2n$ informations (n informations dans la liste de voisins si le graphe est complet, n informations au maximum réparties dans la liste de connexité et la liste de doutes).

Échange, fusion et filtrage

Pour l'instant les seuls événements ayant amené à modifier les listes du capteur c étaient liés à son voisinage direct (un message d'un nouveau voisin est détecté et le voisin est ajouté à N, par exemple). Le balisage est l'action, pour un capteur, de régulièrement partager sa liste de connexité et sa liste de doutes avec ses voisins. Les voisins recevant ce message fusionnent les listes reçues avec les leurs en suivant quelques règles simples.

En l'absence de conflit, les informations du message de l'émetteur sont intégrées à la mémoire interne du récepteur. Par exemple, si c envoie sa liste $C = \{\ldots, (d, 27, e), \ldots\}$ à c' et que ce capteur n'a pas d'entrée relative au nœud d dans ses listes alors on lui ajoute, tout en mettant à jour la source de l'information. c' aura donc comme nouvelle liste $\{\ldots, (d, 27, c)\}$.

En cas de conflits par contre, on se base toujours sur la date t liée à l'information pour déterminer quel capteur à raison et choisir ou non d'intégrer l'information provenant de l'émetteur. Par exemple, si c envoie sa liste $C = \{\ldots, (f, 12, g), \ldots\}$ à c' ayant comme liste $D' = \{\ldots, (f, 16, h), \ldots\}$ alors cela signifie que la présence de f a été mise en doute au temps 16 et l'on ignore l'information de c car on préfère faire confiance aux données les plus récentes.

2 Rendre le réseau connexe

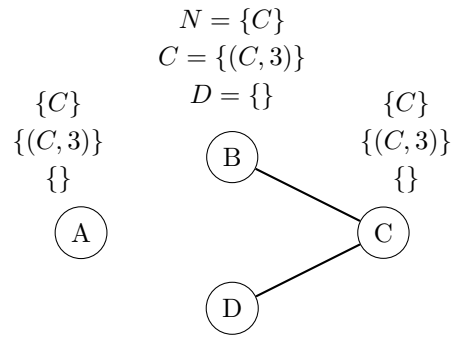
La résolution de ce problème passe par la satisfaction de deux besoins a priori antagonistes. D'une part, il est naturellement nécessaire de réunir les capteurs dans un espace réduit, de façon à ce qu'ils puissent communiquer et échanger des données en continu. D'autre part, et dans l'interêt de leurs utilisateurs, ils doivent s'étaler dans l'espace afin d'effectuer des mesures sur la plus grande superficie possible. Nous sommes donc dans une situation de compromis, dans laquelle une contrainte technique (la portée de communication) force à agglomérer les capteurs en une même zone, tandis que leur but premier est de relever des données en masse et donc, rationnellement, de s'éloigner les uns des autres pour couvrir le plus de terrain possible. La seule issue favorable à ce problème est donc d'aboutir à une situation d'équilibre satisfaisant ces deux contraintes diamétralement opposées.

Le cadre de cet exercice s'accorde parfaitement avec la problèmatique de la prise de décision dans un réseau décentralisé puisque chaque capteur peut être assimilé à un agent autonome, capable d'agir sur la configuration de son environnement en se déplaçant dans l'espace. Quel que soit l'état d'un capteur, la vision du système dont il dispose est locale et doit servir seule à déterminer quelles actions il entreprend. L'objectif est donc ici de proposer une méthode de guidage que chaque capteur peut adopter et qui, par émergence d'une dynamique globale, résoudra notre problème en répartissant les capteurs dans l'espace de façon satisfaisante.

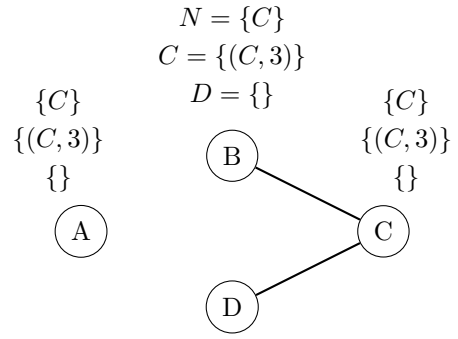
Les boïds de Craig Reynolds [Rey87] sont reconnus pour simuler un comportement de groupe a priori complexe et synchronisé à partir d'un jeu de règles simples. On s'en inspire, ainsi que d'un modèle de mouvement particulaire [CSJ11] se basant sur la répulsion entre molécules de gaz, pour concevoir les trois règles qui gouverneront notre système. Chacune de ces lois, que l'on présente séparément par la suite, va engendrer une influence sur le déplacement de nos capteurs ; influence dont la composition sera détaillée en dernière partie.

Attraction

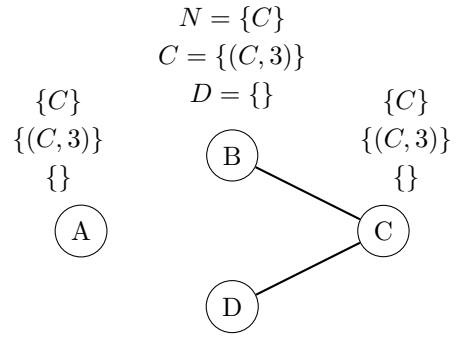
À la lecture de l'énoncé de ce problème, la nécessité de rapprocher les capteurs les uns des autres, afin qu'un réseau de communication ininterrompu se forme, vient naturellement à l'esprit. En effet, la condition *sine qua non* au bon fonctionnement du réseau est la communication. Un capteur isolé est inutile puisque son information n'est pas


$N = \{C\}$
$C = \{(C, 3)\}$
$D = \{\}$
$\{C\}$
$\{(C, 3)\}$
$\{\}$
A
B
C
D
$\{C\}$
$\{(C, 3)\}$
$\{\}$
$\{C\}$
$\{(C, 3)\}$
$\{\}$

(a)


$N = \{C\}$
$C = \{(C, 3)\}$
$D = \{\}$
$\{C\}$
$\{(C, 3)\}$
$\{\}$
A
B
C
D
$\{C\}$
$\{(C, 3)\}$
$\{\}$
$\{C\}$
$\{(C, 3)\}$
$\{\}$

(b)


$N = \{C\}$
$C = \{(C, 3)\}$
$D = \{\}$
$\{C\}$
$\{(C, 3)\}$
$\{\}$
A
B
C
D
$\{C\}$
$\{(C, 3)\}$
$\{\}$
$\{C\}$
$\{(C, 3)\}$
$\{\}$

(c)

FIGURE 2 – ???

partagée.

Par sa loi de la gravitation universelle, Isaac Newton décrit la force qui attire toute paire de corps comme étant proportionnelle à leur masse respective et à la distance les séparant [New87]. Nous nous permettons de simplifier quelque peu son équation et d'en retirer l'aspect massique pour obtenir une règle qui a tout couple de corps associe une attraction proportionnelle à leur distance. S'ils obéissaient à cette loi, nos capteurs auraient naturellement tendance à se grouper, et donc à se mettre à portée de communication.

À l'échelle de notre système, le cadre est différent de celui de la loi de Newtion, et cette influence n'est pas universelle. Nous ne pouvons malheureusement pas réécrire les règles de ce monde et inventer une attirance magique entre toute paire de capteurs. On peut néanmoins la simuler. Si deux capteurs sont à portée et peuvent échanger leur positions respectives alors la détermination de la distance les séparant est aisée. À partir de là, on peut imiter un phénomène d'attration.

Concrètement, on associe à tout capteur c un rayon d'attraction R_a (voir figure 3). Si un capteur voisin se trouve à la fois dans le rayon de communication R_c de c et à l'extérieur de R_a, la force d'attraction que ce capteur devra subir est fournie par la formule suivante :

$$\vec{a} = \frac{\vec{p_c} - \vec{p_v}}{|\vec{p_c} - \vec{p_v}|^2}$$

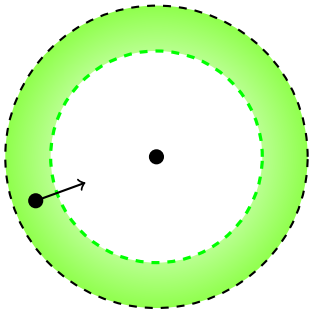


FIGURE 3 – En vert, le rayon d'attraction, dont l'intensité est décroissante de l'extérieur vers l'intérieur.

Dans le phénomène d'attraction réel, l'action est réciproque mais dans notre cas elle est unidirectionnelle. Un capteur est attiré par tous ses voisins mais le processus décrit ne modifie pas directement la position du voisin en question et il n'en subit aucune influence. Il est néanmoins à prévoir que lorsque ce voisin calculera les influences qui s'appliquent à lui, il en subira la réciproque.

On note que l'attraction n'a pas pour but d'approcher des capteurs trop éloignés pour communiquer puisque l'existence d'une connexion entre deux capteurs est un prérequis à l'application de la force d'attraction. Son interêt majeur se révélera en conjonction avec la seconde force présentée.

Répulsion

L'attraction a pour effet d'agglomérer en groupes serrés tous les capteurs démarrant la simulation dans une même zone. Mais même si de cette façon la communication est assurée, les capteurs finissent par tous être superposés sur la même position au bout d'un certain nombre d'itérations et la contrainte de couverture n'est absolument pas satisfaite.

Pour pallier cette déconvenue, on introduit une nouvelle force opposée à l'attraction, la répulsion. Le rayon R_r définit une nouvelle zone radiale qui, contrairement à la loi précédente, expulsera les capteurs envahissants vers l'extérieur. Il est bien sûr nécessaire que $R_r < R_c$. L'influence qui en découle est dirigée vers l'extérieur du capteur envahi et est inversement proportionnelle à la distance les séparant, de façon à engendrer une répulsion plus forte vers le centre [CSJ11] :

$$\vec{r} = \frac{\vec{p} - \vec{p_v}}{|\vec{p} - \vec{p_v}|^2}(R_r - |\vec{p} - \vec{p_v}|)$$

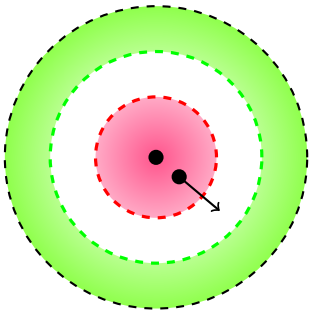


FIGURE 4 – En rouge, le rayon de répulsion, dont l'intensité est décroissance du centre vers les bords.

On prend $R_r = R_a - \varepsilon$, où ε correspond à la largeur d'une bande neutre entourant chaque capteur (voir figure 4). Les capteurs auront naturellement tendance à s'installer dans ces zones libre de toute influence. Nous verrons dans les démonstrations que la conséquence principale de ce positionnement fortement dirigé est un maillage régulier et géométrique. La régularité qu'adopte la structure de notre réseau est un plus à ne pas négliger puisqu'il permet des mesures homogènes par les capteurs.

Gravité

Les deux règles présentées permettent aux capteurs de s'agglomérer en composantes connexes équilibrées mais ces composantes prennent la forme d'îlots et la connexité totale du réseau n'est toujours pas garantie. Dans cette optique, on introduit une dernière influence inspirée du physique, la gravité.

Comme spécifié en introduction, on part de l'hypothèse que les capteurs sont munis de moyens de localisation dans l'espace et nous en avons jusque là allègrement profité pour les calculs de distance nécéssaires aux formules précédentes. Dans la continuité de cette hypothèse, on peut supposer que tous les capteurs connaissent le centre géométrique de l'environnement dans lequel ils évoluent. Cette position peut être la position de laquelle ils ont été largués, une position pré-programmée ou un point calculée de façon façon décentralisée par moyenne de toutes leur positions. Le point important est que cette position est connue de tous et sert d'origine dans leur système de coordonnées.

La gravité est la force qui va attirer tous les capteurs vers le centre de l'environnement. À première vue, une telle règle semble dangereuse car on imagine qu'elle poussera les capteurs à se déplacer vers le centre de la zone au risque de sacrifier l'étendue de la superficie couverte. Néanmoins, on compte sur l'effet combiné des différentes forces, notamment la répulsion, pour résister à cet effet.

L'interêt de la gravité est d'attirer vers un même espace les capteurs isolés et les composantes connexes chanceuses formées par l'attraction et équilibrées par la répulsion. On la calcule de façon uniforme, comme un vecteur unitaire dirigé vers le centre de l'environnement.

$$\vec{g} = -\frac{\vec{p}}{|\vec{p}|}$$

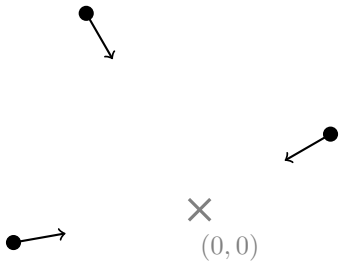


FIGURE 5 – La gravité attire tous les capteurs vers le centre de leur environnement.

Composition d'une force nette

Une fois ces trois forces évaluées, on ne peut pas les appliquer naïvement sur le capteur concerné car il s'agit d'un objet physique soumis à des limitations quant à sa vitesse de déplacement. On associe donc à chaque capteur une vitesse de déplacement maximale qui servira de borne supérieure à la magnitude des trois forces combinées.

Puisque l'on dispose de trois influences différentes, chacune tenant un rôle particulier dans la résolution du problème, l'idée la plus intuitive est d'en déduire une influence moyenne.

$$\vec{f} = \frac{\vec{a} + \vec{r} + \vec{g}}{3}$$

En pratique, on observe que ce choix montre vite ses limites. En effet, sous certaines configurations, deux forces prendront des directions opposées opposées et les combiner de la sorte annulera tout mouvement. Or l'inaction est rarement une solution. Dans d'autre cas encore plus fréquents, les trois forces semblents correctement pondérées : on observe des capteurs isolés flotter lentement vers le centre de leur environnement et s'assembler en composantes connexes de plus en plus grandes. Mais une fois que le maillage est complet et qu'une situation d'équilibre semble avoir été atteinte, le centre du réseau commence à trembler, les capteurs centraux se rapprochent dangereusement et se superposent sur l'origine de l'univers, ignorant totalement la règle de répulsivité. Ce problème est dû au nombre elevé de tensions s'opérant au centre du

réseau, résultant en une pression intense qui fait s'effondrer le réseau jusqu'à ce que tous les capteurs occupent la même position. Moyenner les forces est donc une fausse bonne idée.

On préférera prioritiser les forces, c'est à dire leur donner un ordre d'importance. Le mode opératoire est le suivant :

1. On définit l'ordre des forces, dans notre cas répulsion puis attraction puis gravité.
2. On calcule la force de répulsion et on l'applique en bornant par la vitesse maximale des capteurs.
3. S'il reste de la marge de manœuvre, on recommence avec cette opération avec l'attraction
4. S'il reste de la marge de manœuvre, on recommence avec cette opération avec la gravité

De cette façon, les situations d'urgence telles que l'écartement rapide de deux capteurs trop proches prennent le pas sur des mouvements de composition, tels que celui d'un capteur isolé flottant vers l'origine. Ce procédé est d'ailleurs utilisé pour les boïds afin de prioritiser l'évitement des obstacles face aux forces de cohésion du groupe.

Pour prouver la versatilité de notre système, on propose finalement d'étendre ce modèle de guidage afin de permettre aux capteurs d'éviter automatiquement les obstacles par l'ajout d'une nouvelle force virtuelle. Ici, un obstacle est défini comme étant une zone non traversable. En réalité, il peut s'agir d'un bâtiment, d'une zone dont les relevés ne nous intéressent pas, ou d'un espace interdit au appareils volants.

Dans un soucis de simplicité (et de délais !), on limite les obstacles à des formes circulaires afin de mieux s'accorder à l'aspect radial des différentes forces introduites jusqu'à présent.

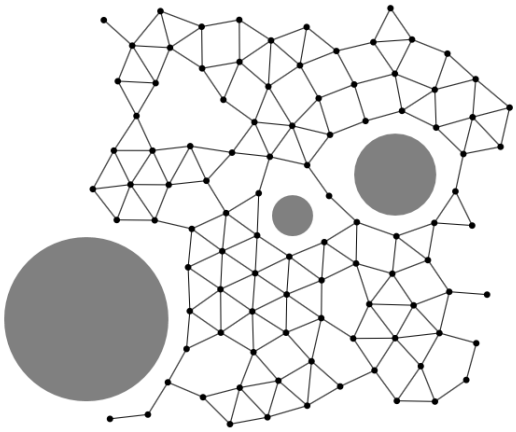


FIGURE 7

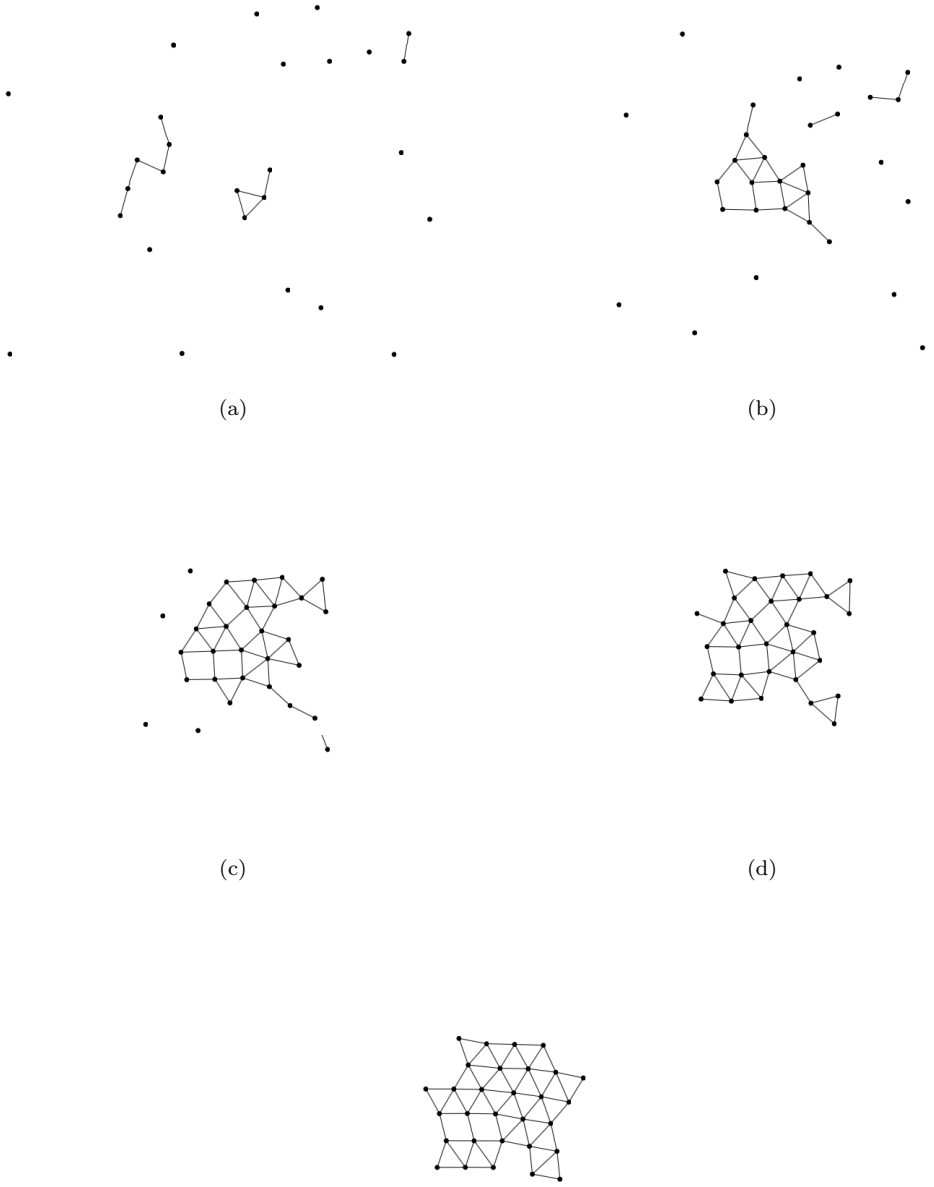
Conclusion

Références

[CSJ11] Teddy M. Cheng, Andrey V. Savkin, and Faizan Javed. Decentralized control of a group of mobile robots for deployment in sweep coverage. *Robotics and Autonomous Systems*, 59(7–8) :497 – 507, 2011.

[New87] Isaac Newton. *Philosophiæ Naturalis Principia Mathematica*. 1687.

[Rey87] Craig W. Reynolds. Flocks, herds, and schools : A distributed behavioral model. In *SIGGRAPH '87 : Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34. ACM, 1987.



(a) (b)

(c) (d)

(e)

FIGURE 6 – Différentes étapes de la construction d'un réseau connexe.