Université Libre de Bruxelles

INFO-H-414 - Swarm Intelligence

Swarm Robotics Project: Chain formation strategy

Author:
Jacopo De Stefani

Supervisors:
Prof. Marco Dorigo
Prof. Mauro Birattari

July 29, 2013

Contents

Introduction

The main objective of the project is to develop a chaining strategy for a swarm of robots in an indoor structure, that connects the nest (i.e. the area where the robots are initially located), to five target location spread across the environment (identified by the black dots).
The environment is composed by open corridors, as well as obstacles, like walls and cubic structures (that could represent abstractions for furniture in the the closed environment).
The simulated robots are the *s-bots* [3], developed at the EPFL within the framework of the EU-funded Swarm-bots project.
With respect to the real robot configuration, only a limited set of sensors and actuators are used in the project.
To be more precise, robots are able to sense the presence of obstacles in the short range (4-20 cm, by means of IR proximity sensors or short range ultrasonic distance scanner) and in the long range (30-150 cm, by means of long range ultrasonic distance scanner) and detect a variation in the ground color (by means of the IR ground sensors).
Furthermore, they can move around in the environment by means of a pair of treels (a combination of track and wheels), each one connected to differential drive motor.
Robots can communicate through the range and bearing system, that allow them to broadcast a message to neighboring robots and localize the messages sent by other agents in terms of range (receiver-sender distance) and horizontal and vertical bearing (angular displacement of the sender with respect to a reference point on the receiver).
One basic principle in Swarm Robotics is that a collective behavior could emerge from local interactions between the robots and with the environment.
In this project I will develop a controller for a single robot, which will be executed in parallel by all the robots in the swarm, and observe the behavior emerging at the group level.

Controller overview

General structure

The robot controller has been structured according to the sense-think-act paradigm.
In fact, at each time step, the robots will:

1. *(Sense)* - Read the informations collected by the available sensors.
2. *(Think)* - Determine the values to send to the actuators according to the state machine defined in Controller details and the information from the sensors.
3. *(Act)* - Control the actuators.

In terms of code:

1. *(Sense)* - The *ParseX* function are used to read the values of the different sensors (proximity, ground, distance scanner and Range and Bearing) and provide information to the following step in a suitable form (e.g. repulsion vector or beacon table).
2. *(Think)* - The core of the behavior is implemented as a finite state machine (FSM), where each state of the automaton is implemented as function. In this way the controller of the robot just need to execute the function corresponding to the agent's current state.
3. *(Act)* - According to the position, and the control values computed in the previous step, determine the speeds to actuate on the wheels and the information to broadcast using the Range and Bearing System.

Vector fields

According to the principles of the swarm robotics, a behavior at the swarm level should emerge from local interactions at the robot level, using information either coming from the environment or from other robots. These interactions have been modeled using a potential-field approach (cf.[2]), based on the readings from the sensors. These virtual-potential fields U_i are defined on the whole environment and robots are able to compute locally the force $\mathbf{F}_i$ resulting from the interaction with the corresponding field as:

$$\mathbf{F}_i = -\nabla U_i \tag{1}$$

With this approach, it is possible to explicitly construct a field U_i to induce a certain behavior on the robot. For this purpose, the following virtual fields have been defined:

- Obstacle avoidance - U_{obs}
- Distance scanner - U_{ds}
- Range and Bearing - U_{rab}

The resulting forces from the aforementioned vector fields are determined from the corresponding sensor readings. That is, each reading is transformed into a vector whose angle and magnitude depends on the value of the reading itself. All the vector are expressed in polar coordinates, in the form (magnitude,angle). Note that $\mathbf{F_{obs}}$ and $\mathbf{F_{obs}}$ are repulsive, while $\mathbf{F_{rab}}$ is attractive.

Obstacle avoidance

$$\mathbf{p_i} = (r_{prox_i}, \theta_i) \tag{2}$$

where r_{prox_i} corresponds to the value of the reading of the i^{th} proximity sensor and θ_i is the corresponding angle.

$$\mathbf{F_{obs}} = (1.5, \theta_{sp}) \ \mathbf{sp} = \sum_i -\mathbf{p_i} \tag{3}$$

where θ_{sp} corresponds to the angle of **sp** vector.

Distance scanner

$$\mathbf{ds_i} = (1 - \frac{150 - r_{ds_i}}{150 - 4}, \theta_i) \tag{4}$$

where r_{prox_i} corresponds to the value of the i^th reading of the distance and θ_i is the corresponding angle. The value of the distance scanner reading is normalized in the range $(4[cm], 150[cm])$ (Lower bound short range reading, upper bound long range reading). By subtracting the normalized value to 1, one is able to obtain a vector whose magnitude is inversely proportional to the distance from the obstacle.

$$\mathbf{F_{ds}} = (1.0, \theta_{sds}) \ \mathbf{sds} = \sum_i -\mathbf{ds_i} \tag{5}$$

where θ_{sds} corresponds to the angle of **sds** vector.

Range and Bearing

$$\mathbf{F_{rab}} = (1.0, \theta_{rab}) \tag{6}$$

where θ_{rab} corresponds to the angle of RAB reading.

General idea

The general idea of the method is that the robots should first quit the initial deployment room, characterized by four surrounding walls, one of them containing an opening to let the robots move outside, and a dark grey floor, to then start the exploration phase required to find the target spots.
In order to reduce the interference phenomenon at the exit of the nest, a sequential deployment mechanism based on the robot id has been developed.
After exiting the nest, the agents should explore the environment, either individually (as *explorers*) or using the collectivly-gathered knowledge of the environment (i.e. the chain).
The exploration of the environment should, in principle, profit of the already available information. Nevertheless, this method achieves the required chain formation behavior using only the proximity sensors and the distance scanner.
If no information is available, the robot could decide (stochastically) to become a starting point for a new chain in the environment.

Rules The actual chain formation behavior is driven by five simple rules:

1. If the **nest** has been **left**, and **no** chain beacon have been already **sensed**, after t_{ns} time step, decide with probability p_{btoe} to stop and become a chain end.
2. If the **nest** has been **left**, and **exactly one** chain beacon has been **sensed** at a distance greater than d_{chain}, stop and become a chain end.
3. If a **chain end** has **more than one** neighboring **beacon**, but **less than three**, then it changes its state to *Chain member*.
4. If a **chain member** has **more than two** neighboring **beacons**, then it changes its state to *Chain junction*.
5. The chain identifier of a new beacon is determined by incrementing of one unit the chain id of the closest beacon.

Chain structure The chain will then be composed by three different kind of robots:

- **(E) - Chain end:** Any robot connected to at most one other agent.
- **(M) - Chain member:** Any robot connected to exactly two agents.
- **(J) - Chain junction:** Any robot connected to more than two agents.

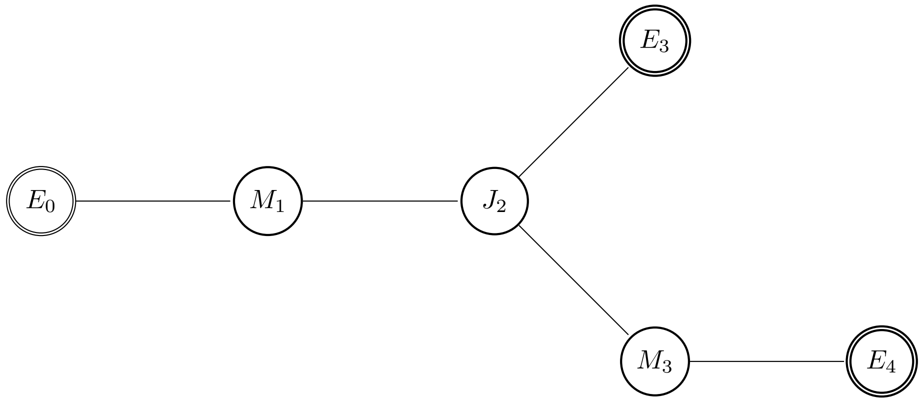


Figure 1: Chain example with nodes labeling and id

References

Concerning the chain formation rules, the main sources of inspiration were [5], [4] which helped me to better understand the chain structure (in particular, how to distinguish elements in the chain) and the chain navigation behavior (which have not been implemented here) and [1] for defining the conditions to extend the chain.
Furthermore, the idea of an incremental deployment has been taken from [6]. While in [6] the incremental deployment was used to limit energy consumption in the deployment phase, here the same idea is used to prevent the interference phenomenon that could occur at the nest exit. In fact, whenever a relevant number of robots (10+) tries to exit the nest at the same time, each agent will spend more time performing obstacle avoidance with respect to one another, rather than actually exiting the nest.

Controller details

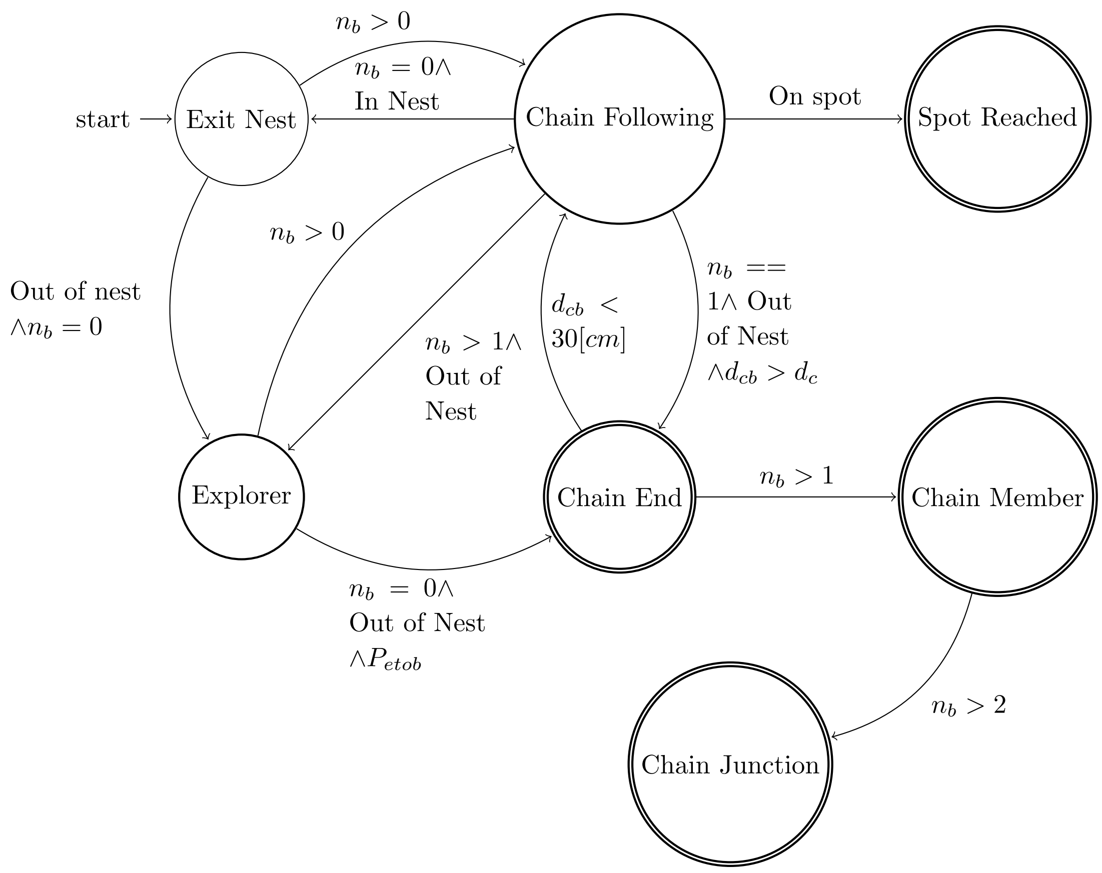


Figure 2: Implemented FSM for the *s-bot* chain formation behavior

Controller parameters

Parameter	Description	Value
v_{wheel}	Wheel velocity (straight movement)	$10[\frac{cm}{s}]$
P_{etob}	Probability of starting a chain while being an explorer	0.05
d_{chain}	Minimum distance among elements in the chain	130 [cm]
ω_{ds}	Angular rotation velocity of the distance sensor	$2\pi[\frac{rad}{s}]$
t_{ns}	Delay time on the decision to stop outside the nest	100 [steps]

Table 1: Controller parameters overview

Resulting forces

States details

Exit nest

$$\mathbf{F_{res}} = \mathbf{F_{obs}} + \mathbf{F_{ds}} + \mathbf{F_{rab}} \tag{7}$$

The exit nest state is the initial state of the robots, which is maintained as long as it remains in the nest. The robots uses the distance scanner and proximity sensors to find their way out of the nest. In addition to those forces, the robot is also attracted (by means of $\mathbf{F_{rab}}$) towards the closest robots in the RAB sensing range that are in the *Explorer* or the *Chain following* state. This is done beacause such kind of robots are either directing or already outside of the nest, thus following them will guide a robot in the right direction. If a robots senses any kind of beacon it changes its state to *Chain following* Otherwise, it passes in *Explorer* state.

Explorer

$$\mathbf{F_{res}} = \mathbf{F_{obs}} + \mathbf{F_{ds}} + \mathbf{F_{str}} \tag{8}$$

In the *Explorer* state, if no beacon is sensed, the robot moves outsides the nest without making any decision for t_{ns} time steps, then deciding, at each simulation step, whether to stop or not.
The decision is stochastic and occurs with probability P_{etob}.
Otherwise, the robot goes straight ($\mathbf{F_{str}} = (1.0, 0)$).
In any case, if the agent goes back to the nest, it enters the *Exit Nest* state.

Chain following

$$\mathbf{F_{res}} = \mathbf{F_{obs}} + \mathbf{F_{ds}} + \mathbf{F_{str}} \tag{9}$$

The *Chain Following* state simply consists of a random walk of the robot in the environment.
The robots normally goes straight (guided by $\mathbf{F_{str}} = (1.0, 0)$).
Its direction is then modified by the perceived obstacles both in a short range ($\mathbf{F_{obs}}$) and in a long range ($\mathbf{F_{ds}}$).
As soon as the distance of a robot from the closest beacon is greater than the minimum chain distance ($d_{cb} > d_c$) and exactly one beacon is sensed nearby ($n_b == 1$) (cfr. Rules (2)) the robots stops and becomes a *Chain End*.
If the robot loses contact with a chain beacon, it turns itself of 180 degrees. before restarting exploration.
In any case, if the agent goes back to the nest, it enters the *Exit Nest* state.

Chain End

$$\mathbf{F_{res}} = (0, 0) \tag{10}$$

The *Chain End* state is reached when a robot connects himself to the current end of the chain, becoming thus the new termination.
In this state, the robot stands still and broadcast information concerning its state, id and chain_id to the neighboring robots.
If any beacon is sensed within 30 cm the robot starts moving again, in order to find a better position where to attach in the chain.
The transition to the *Chain Member* state occurs anytime another agent connects to the current one.

Chain Member

$$\mathbf{F_{res}} = (0,0) \tag{11}$$

A *Chain Member* only acts as a beacon, broadcasting state information and acting as a relay for the back-propagation of the *Target found* message. A *Chain Member* robot becomes a *Chain Junction* if another robot decides to join the chain by attaching to it.

Chain Junction

$$\mathbf{F_{res}} = (0,0) \tag{12}$$

The sole purpose of a *Chain Junction* is to differentiate himself from the other beacons and stop the back-propagation of the *Target found* message.

Results

Metric definitions

The developed method will be characterized by two metrics:

1. Number of robots in chain (counted using the `robot.in_chain` variable)
2. Completion time (i.e. time needed to form a chain connecting all the nest to all the 5 targets present in the environment).

The metrics have been measured on 50 experiments, each one of them was run using a randomly generated seed using the `RANDOM` environment variable predefined on the Bash shell. Each second is simulated by 10 simulation steps. Since the strategy is mainly based on a random exploration, its performance can be regarded as a baseline value for comparison with more advanced methods.

Communication range influence

The robots communicate among themselves by means of the Range and Bearing system, whose range is left to the default value given in the configuration file ($150[cm]$).
In the method, the main functions of this communication system are:

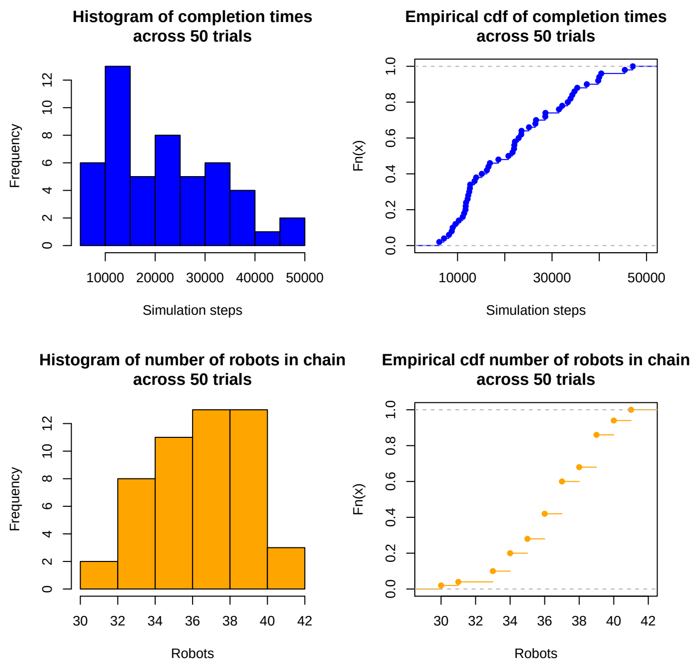
- Attract the robots outside the nest (cf. Exit nest)
- Determine the attaching point to the chain (cf. Chain following)

By reducing the communication range one would expect that:

- An higher number of robots would be required to form the chain (since the robots would be closer to each other).
- The process of exiting the nest would be slower (since less neighboring robots will be sensed).

On the other hand, since the chain following behavior consists of a random walk, I believe that the reduction of the communication range would not affect this component of the behavior.
Unfortunately, an extensive study on the effect of the communication range, and thus the scalability of the method hasn't been performed within this project.

Results distribution



(a)

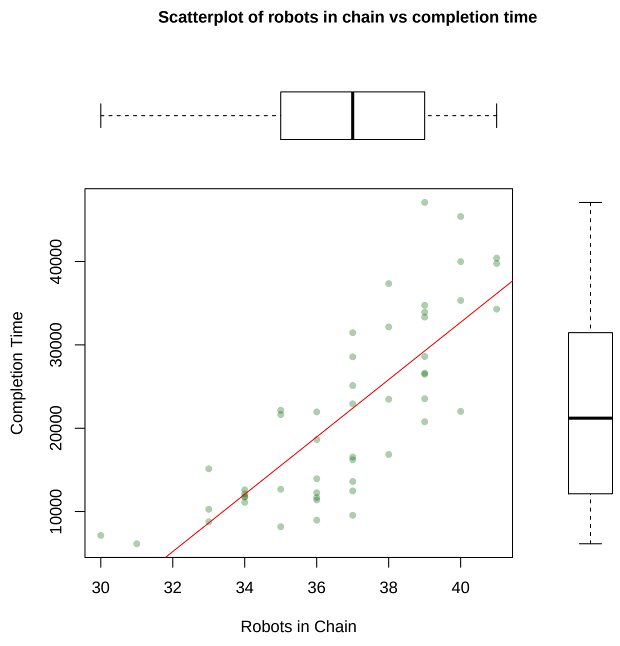
	Robots in Chain	Completion Time
Mean	36.82	21779.52
Std Dev	2.5689671	11138.0954867
CV $= \frac{\sigma}{\mu}$	0.06952743	0.51140223
Median	37	21209.5
Min	30	6133
Max	41	47097

(b)

Figure 3: Panel containing the histograms and the empirical cumulative distribution functions for the designed metrics (a) with a summary of the relevant statistics of the two distributions (b)

The analysis of the distribution of these metrics shows that, in the 90% of the experiments, the number of robots required to form a chain is bounded in the interval $[32, 40]$.
The variability of this results can be explained by the nature of the targets and their placement in the environment. In fact, since the spots can only be sensed if a robot is completely on it, sometimes it may happen that a robot ignores the presence of the spot, even while being next to it, and tries to extend the chain in another direction, with respect to the target.
On the other hand, even though approximately the 75% of the trials are completed within 30000 simulation steps, there is a great variability in the time required to successfully complete the task, as shown by the variation coefficient (CV).
Given the nature of the method, the change in the value of the seed will affect only the initial deployment position and rotation and the stopping position of the first robot (cf. Controller details).
One can then conclude that the random change in the seed has a relevant impact on the completion time, while having a minor influence on the number of robots in the chain, that can be partially explained by the nature of the environment.

Scatterplot and correlation analysis



(a)

	Value	P-Value
Pearson - r	0.7934599	6.357e-12
Kendall - τ	0.6691023	5.554e-11

(b)

Figure 4: Enhanced scatter plot of the completion times as a function of the number of robots (a), with the results of the correlation analysis (b)

Intuitively, one may assume that the completion time is positively correlated with respect to the number of robots in chain (since more robots require more time to be deployed).
This intuition is confirmed by the graphical representation of the completion time as a function of the number of robots in the chain, along with the representation of a linear model fitting the points obtained through the experiment as well as by the values of the correlation coefficients, and their statistical significance (attested by a p-value considerably small than the confidence level $1 - \alpha = 0.05$.

References

[1] Simon Goss and Jean-Louis Deneubourg. Harvesting by a group of robots. In *Proceedings of the First European Conference on Artificial Life*, pages 195–204, 1992.

[2] Andrew Howard, Maja J Matarić, and Gaurav S Sukhatme. Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *Distributed Autonomous Robotic Systems 5*, pages 299–308. Springer, 2002.

[3] Francesco Mondada, André Guignard, Michael Bonani, D Bar, Michel Lauria, and Dario Floreano. Swarm-bot: From concept to implementation. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1626–1631. IEEE, 2003.

[4] Shervin Nouyan, Alexandre Campo, and Marco Dorigo. Path formation in a robot swarm. *Swarm Intelligence*, 2(1):1–23, 2008.

[5] Shervin Nouyan and Marco Dorigo. Chain formation in a swarm of robots. *IRIDIA, Université Libre de Bruxelles, Tech. Rep. TR/IRIDIA/2004-18*, 2004.

[6] Timothy Stirling and Dario Floreano. Energy-time efficiency in aerial swarm deployment. In *Distributed Autonomous Robotic Systems*, pages 5–18. Springer, 2013.