Experiments in swarm robotics

– Intelligent Robotic Systems –

Andrea Roli
andrea.roli@unibo.it
Dept. of Computer Science and Engineering (DISI)
Alma Mater Studiorum Università di Bologna

Aggregation task

A typical task in SR is *aggregation*, i.e. the swarm behaviour consisting in gathering in one place. A typical method for aggregation in SR comes from the observation of social insects. An informal model for aggregation is described as follows:

- Each robot performs a random walk (possibly with collision avoidance) and stops with probability P_s;
- P_s depends on the number of nearby robots already stopped;
- the higher this number, the higher P_s (positive feedback);
- when a robot has stopped in a group, it can leave it with a probability P_w and start again to wander;
- the higher the number of stopped nearby robots, the lower P_w (robots at the boundary of the group perceive less neighbours, therefore their probability of leaving the group is higher than that of those inside. This introduces a negative feedback in the dynamics of the swarm).

Robots can perceive their nearby companions by means of a *range and bearing communication system*, which will be described later.

Let $S \in [0,1]$ be the spontaneous stopping probability, $W \in [0,1]$ the spontaneous walking probability and N the number of nearby stopped robots. These probabilities can be updated as follows:

- $P_s = \min\{P_s^{max}, S + \alpha N\}$
- $P_w = \max\{P_w^{min}, W - \beta N\}$

where S, W, α, β, P_s^{max} and P_w^{min} are parameters of the control software. These parameters in general depend upon the number of robots, the size of the arena and the maximal range used for the range-and-bearing. For a rectangular arena of 4×5 meters, 50 robots and a maximal range of 30 cm, you may initially try with: $W = 0.1$, $S = 0.01$, $P_s^{max} = 0.99$, $P_w^{min} = 0.005$, $\alpha = 0.1$, $\beta = 0.05$.

Range and bearing system

(This is a full description of the device.[1] For an operational use, you can skip this paragraph and move to the next examples)

The range-and-bearing system allows robots to perform localized communication. Localized communication means that a robot, upon receiving data from another robot, also detects the position of the sender with respect to its local point of view. It is important to notice that the range-and-bearing system is not like WiFi. First, because two robots can exchange data only if they are in direct line of sight — if an object is between two robots, the robots can't communicate. Second, because robots that send data can only broadcast it in a limited area — you can't pick who you talk to as you would with an IP address. Third, the robots can exchange only 10 bytes of data. To set the data to broadcast, use `set_data()`. This function accepts input in two forms. You can write `set_data(idx, data)`, and this means that you set the idx-th byte to the value of `data`. `data` must be an integer number in the range [0,255]. Alternatively, you can write `set_data(data)`, where `data` must be a table containing exactly 10 integer numbers in the range [0,255]. At each time step, a robot receives a variable number of messages from nearby robots. Each message is stored in a table composed of `data` (the 10-bytes message payload), `horizontal_bearing` (the angle between the robot local x axis and the position of the message source; the angle is on the robot's xy plane, in radians), `vertical_bearing` (like the horizontal bearing, but it is the angle between the message source and the robot's xy plane), and `range` (the distance of the message source in cm).

[1] From https://www.argos-sim.info/plow2015/

In your solution, you may want to use the following pieces of code.

```
-- send something to signal your presence (not moving) to the other robots
robot.range_and_bearing.set_data(1,1)
```

meaning: on byte 1 (first argument) send value 1 (second argument).

```
-- send something to signal to the other robots that you are moving
robot.range_and_bearing.set_data(1,0)
```

meaning: on byte 1 (first argument) send value 0 (second argument).

```
-- Count the number of stopped robots sensed close to the robot
function CountRAB()
  number_robot_sensed = 0
  for i = 1, #robot.range_and_bearing do
    -- for each robot seen, check if it is close enough.
    if robot.range_and_bearing[i].range < MAXRANGE and
                    robot.range_and_bearing[i].data[1]==1 then
      number_robot_sensed = number_robot_sensed + 1
    end
  end
  return number_robot_sensed
end
```

`MAXRANGE` can be initially set to 30 (cm). Test the behaviour also with other values.

In ARGoS you can use the RAB sensor mounted on the footbots:
`<range_and_bearing implementation="medium" medium="rab" />`

By default, a message sent by a foot-bot can be received up to 3m. By using the `rab_range` attribute, you can change it to e.g. 1m as follows:

```
<foot-bot id="fb" rab_range = "1">
```

Exercises

Three exercises of different aggregation behaviours are suggested. Only the first is mandatory, so focus on this and, if you have time, proceed with the following ones.

Exercise 1

Implement the aggregation behaviour according to the model described above. Of course, you can also implement your own aggregation mechanism. Experiment with different values for S, W, α and β. The behaviour can be modelled by means of a probabilistic automaton.[2] Update probability rules:

```
Ps = math.min(PSmax,S+alpha*N)
Pw = math.max(PWmin,W-beta*N)
```

For implementing the probabilistic decision mechanism—based on a Bernoulli distribution with probability p—you may want to use a piece of code like the following:

```
t = robot.random.uniform()
if t <= p then
  ...
else
  ...
```

Exercise 2: Aggregation on a black spot

To build an aggregate on a dark spot we may change the probability update rules as follows:

- $P_s = \min\{P_s^{max}, S + \alpha N + D_s\}$
- $P_w = \max\{P_w^{min}, W - \beta N - D_w\}$

where D_s and D_w are the probabilities of stopping (or remaining) on the black spot.

[2]Suggestion: to visually inspect the overall behaviour, use LEDs coloured depending on the state of the automaton.

Exercise 3: Aggregation on one out of two black spots

In nature, if there are many places in which insects can aggregate, they collectively choose only one. Try the behaviour designed in the previous exercise in an arena with two black areas and observe the behaviour. Do the robots achieve a consensus and stop on the same spot (at least the majority of the robots)? Try to describe the behaviour of this swarm as the evolution in time of a dynamical system.

Food for thought

- Is the behaviour robust wrt parameter values?
- What are the advantages and disadvantages of a collective choice?
- How would you bias the aggregation on a specific area of the arena?
- What are pros and cons of distributed solutions wrt centralised ones?