FINAL PROJECT
REAL TIME

SCUOLA SUPERIORE SANT'ANNA

DEPARTMENT OF ENGINEERING

Ants

Authors:
Federico Dragoni
Leonardo Lai

Email:
fe.dragoni@sssup.it
l.lai@sssup.it

Date: December 23, 2018

1 Abstract

The project simulates a colony of ants looking for food in a field. The search is guided by pheromones, released by ants themselves when they find something interesting. This communication mechanism, which is a remarkable example of *stigmergy* (1), allows ant to discover paths between their anthill and food sources, and possibly converge towards the an optimal one (2).

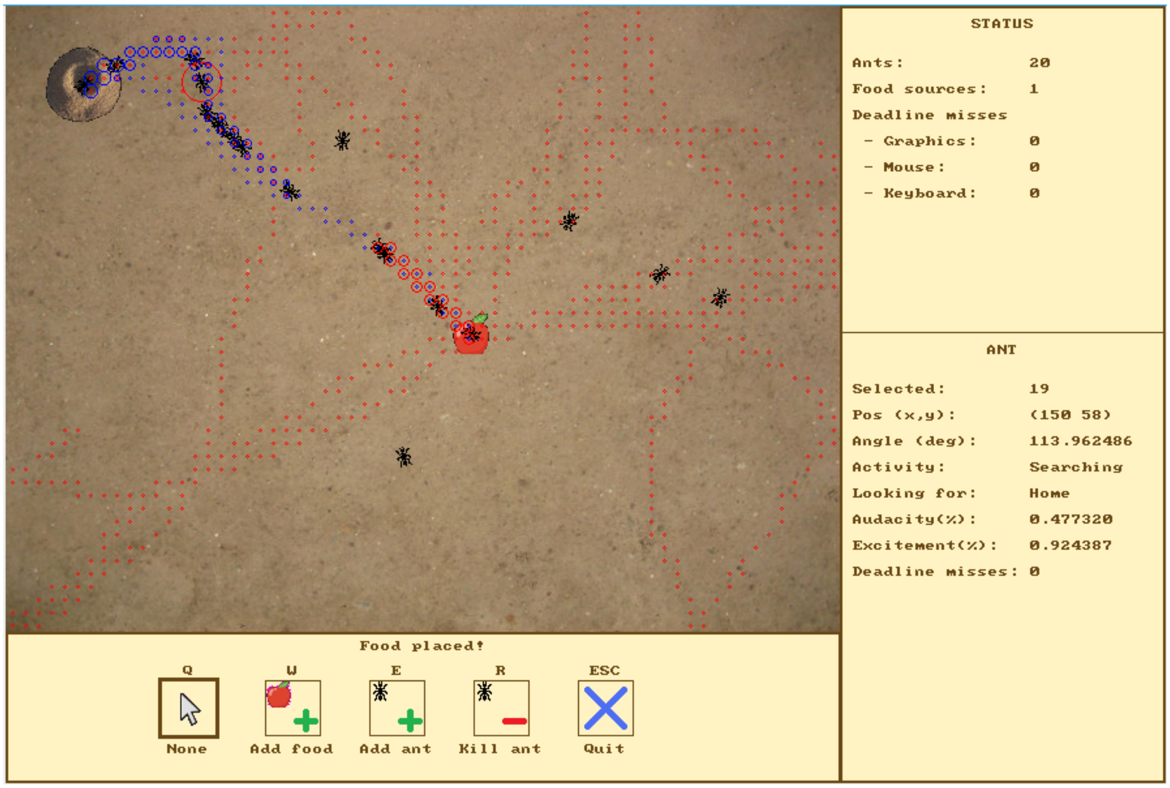

Figure 1: Overview of the application

2 Description

2.1 Field

The **field** is represented as a two-dimensional rectangular box. Every point is identified by a unique pair of natural coordinates (x, y). Moreover, the field is split in a dense grid of square cells, each storing local information about the pheromones laid on the cell itself. Cell size can be customized: choosing a finer granularity results in increased model accuracy, but also greater computational cost (pheromones updating operations are expensive).

2.2 Ant

The field is populated by a colony of **ants**, whose population size can be controlled by the user. Every ant evolves autonomously, yet its decisions might be affected by other individuals by the means of pheromones. Considering one ant, its high-level behaviour is quite simple: it moves out from the anthill (home), then starts looking for food; when any is found, the ant reaches it, has a meal and eventually goes back to home. After resting for a while, the ant repeats again the same routine. When the ant is searching, it can be either exploiting available information or exploring.

Exploiting
Exploiting means that the ant is currently following a trail of pheromones or actively looking for one.

Exploring
Exploring means that the ant is looking for new paths, ignoring pheromones.

Most of the times the ant is exploiting, but may switch to exploring when of these two situations occurs: a local maximum is detected in a pheromones trail, or the ant autonomously decides to explore. The former prevents creatures from getting stuck in local optima cells, where no object actually exists but, for some reason, it has the highest pheromones values among the neighbourhood, thus the gradient can't suggest the next direction. The latter, instead, allows them to discover new possibly better paths even when a stable route has been found and stabilized. The probability of switching from exploiting to exploring has to be kept quite low, otherwise pheromones become almost useless and, consequently, the search becomes random. Exploring state lasts for a finite number of ticks, after which the ant returns exploiting. Fig. 2 summarizes what said above.

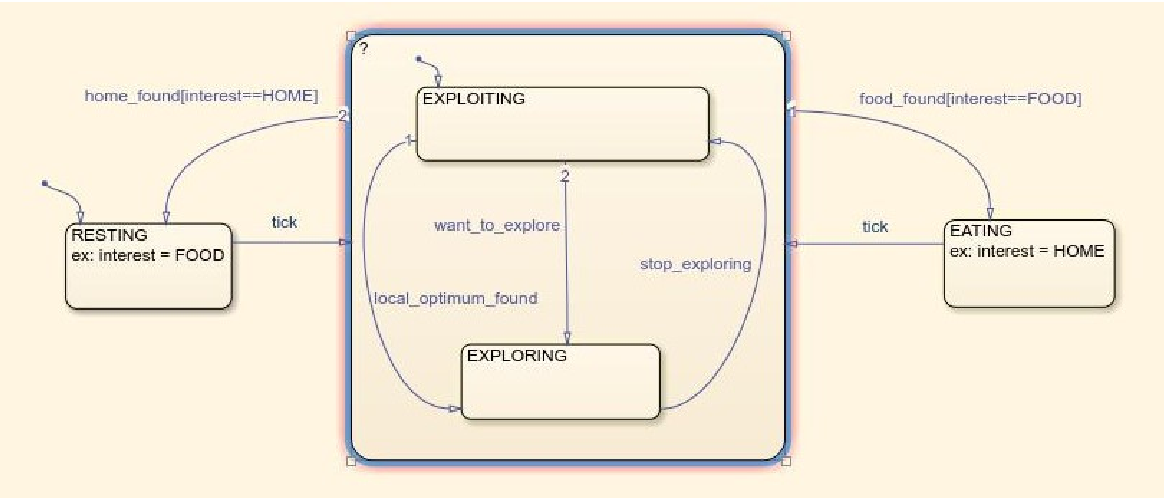

Figure 2: Ant routine overview

Ants cannot escape from the field: whenever they reach a border, they change direction as if bounced by a wall.

In absence of pheromones nearby (i.e. within detection radius), the ant moves in a pseudo-rectilinear fashion, which is hybrid between a perfectly linear trajectory and a random-walk (3). Basically, it keeps going straight with orientation (θ) but with slight random deviations, whose angle ($d\theta$) is uniformly distributed in a small range (c = exploration cone parameter).

$$d\theta_k \in \mathcal{U}\left(-\frac{c}{2}, \frac{c}{2}\right)$$
$$\theta_{k+1} = \theta_k + d\theta_k$$

With respect to one ant, a pheromone is regarded to be nearby if the corresponding cell has non-zero value and its distance from the ant (considering the centres of cell and ant, respectively) is less than or equal the olfactory detection radius.
When one ant smells anything, and if it is interested in that particular fragrance (see below), it stops wandering pseudo-randomly and starts moving towards the pheromone location. If the scent comes from multiple directions, that is the most likely case, the ant picks the one with highest intensity. In this way it will almost always end up following the pheromones trail left by those predecessors who made it through the search.

2.3 Pheromones

Pheromones exist in two different fragrances: food and home. Ants are interested in finding pheromones of their target type, and start releasing them immediately after reaching the desired object, so that other fellows can possibly smell it and find the target quicker. For instance, if an ant is currently looking for food, its behaviour will be affected by any pheromone of type food it encounters; when that ant eventually finds actual food, it starts releasing pheromones of type food.
The presence of pheromones is not enough alone to determine the true location of food: an intensity value has to be associated to each pheromone in a cell. Two distinct mechanisms affect this value: **evaporation** and **drop discount**.

Evaporation
Evaporation reduces the pheromones intensity over time after their release. In this way, the information becomes non-persistent, and the smell is bound to disappear unless another ant refreshes it. Suboptimal paths are likely to vanish, instead those which are trodden by many ants will hopefully be preserved. The pheromone stored in cell i decays according to a geometric series:

$$p_{k+1}^{(i)} = \alpha p_k^{(i)}$$

Drop discount
Ants do not release pheromones uniformly, but the intensity value gradually diminishes as the ant walks away from its previous target. As a consequence, an ant will almost stop dropping pheromones if a lot of time has elapsed since the last time it had visited anthill or found food. Conversely, fresh information is associated to

high pheromones levels. The dropped quantity r is initially R and gets discounted by factor γ after each step.

$$r_0 = R$$
$$r_{k+1} = \gamma r_k$$

Evaporation and drop discount together determine a gradient in the odor trails, and make it possible for ants to identify the smell direction. It is necessary that $\alpha > \gamma$, otherwise evaporation obfuscates drop discount and the mechanism stops working because the gradient verse is flipped.

3 Interface

The interface is split into three areas: field, toolbar and status monitor.

3.1 Field

The field features a dirt background (Allegro bitmap), on which ants (Allegro sprites) can move and release pheromones. The latter are shown as small circles, whose colour depends on the fragrance (red is food, blue is home) and radius is proportional to the intensity. Similarly, food sources (sprites) decrease in size as ants consume them.

3.2 Status monitor

The status monitor, visible on the right of the window, displays the number of ants and food resources currently being in the field. As well, it the number of deadline misses of graphics, mouse and keyboard tasks.(Fig. 3). When you click on an ant, the main information about the selected ant are plotted on an additional monitor right below: position and orientation in the field, current activity, audacity and excitement levels, number of deadline misses by its task so far.

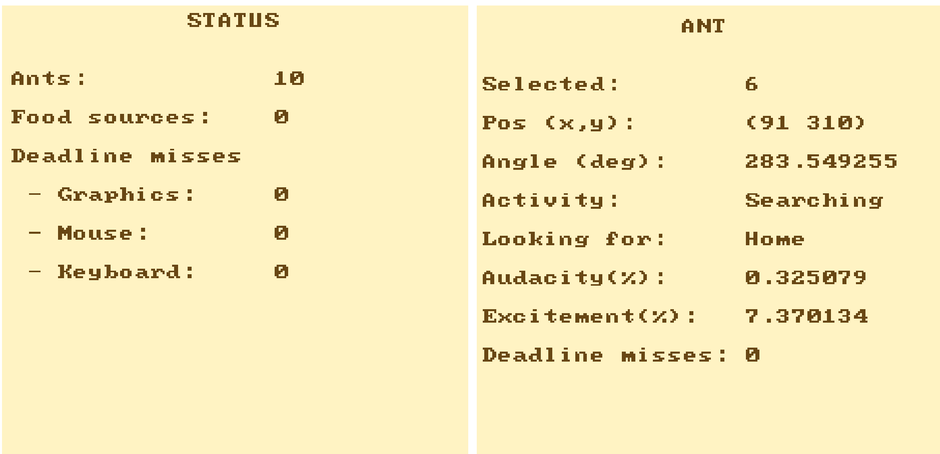

Figure 3: Status monitor

3.3 Toolbar

The toolbar, visible in the bottom part of the window, allows the user to switch between 5 different modes by pressing the dedicated key (Fig. 4). The button name is always shown below the corresponding icon, whereas a more detailed description about the current mode (including usage details) appears right above the toolbar. Here follows the list of all the possible modes:

- Q [Select]: left-click on an ant to display on the right panel status and statistics about the selected ant itself.
- W [Add food]: left-click on the ground to create "food" in the desired position.
- E [Add ant]: left-click to spawn a new ant from the anthill.
- R [Kill ant]: left-click on one ant to kill it (gracefully).
- ESC [Quit]: abort the simulation and close the program

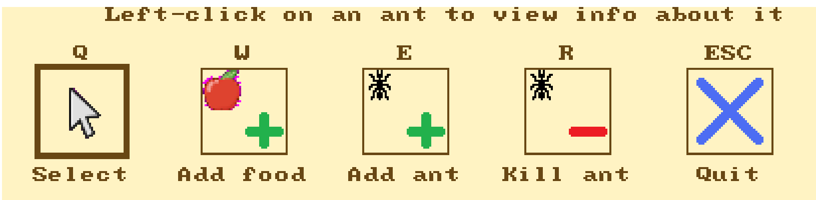

Figure 4: Toolbar

4 Code overview

4.1 Programming language and libraries

The application is entirely developed in C, using the libraries Allegro 4.4 (4) and pthread (5). The project root directory has four sub-folders: `conf` contains the configuration files, `docs` the documentation, `img` the images used by the application and `src`, the most important one, stores all the source files. Two additional folders, `bin` and `build`, are created at compilation time and contain object and binary files, respectively.

4.2 Data structures

The state of the entities involved in the application is represented by few data structures. Here is a list of the most important ones:

- `ant` (declared in `ants.h`) is the ant descriptor storing all the info about the current state and behaviour of the tiny creatures. Each ant has its own descriptor.
- `ants` (declared in `ants.h`) is a container for all the ant descriptors.

- `food` (declared in `field.h`) is the descriptor of any food source in the field.
- `foods` (declared in `field.h`) is a container for all the food descriptors.
- `cell` (declared in `field.h`) stores information about the pheromones dropped by ants within a cell of the grid.
- `ph` (declared in `field.h`) is the grid of pheromones cells.
- `task_par` (declared in `rt_thread.h`) is the descriptor of any real-time thread, containing information about the code, arguments and scheduling parameters.

4.3 Tasks

Multiple real-time threads run concurrently during the simulation. Three threads are allocated for refreshing the graphics, handling keyboard and mouse inputs, respectively. Another thread updates the pheromones over time according to the evaporation law. Also, each ant has its own thread, which controls the evolution according to the aforementioned model.

- Keyboard task allows switching between the different modes
- Mouse task enables interaction between the user and the environment
- Evaporation task handles the pheromones decay
- Ant tasks controls the motion and behaviour of the respective ant

The default configuration of scheduling parameters is shown in Table 1.

Task	Period	Deadline	Priority
Graphics	50	50	30
Keyboard	200	200	10
Mouse	50	50	20
Ants	50	50	20
Evaporation	50	50	20

Table 1: Tasks parameters

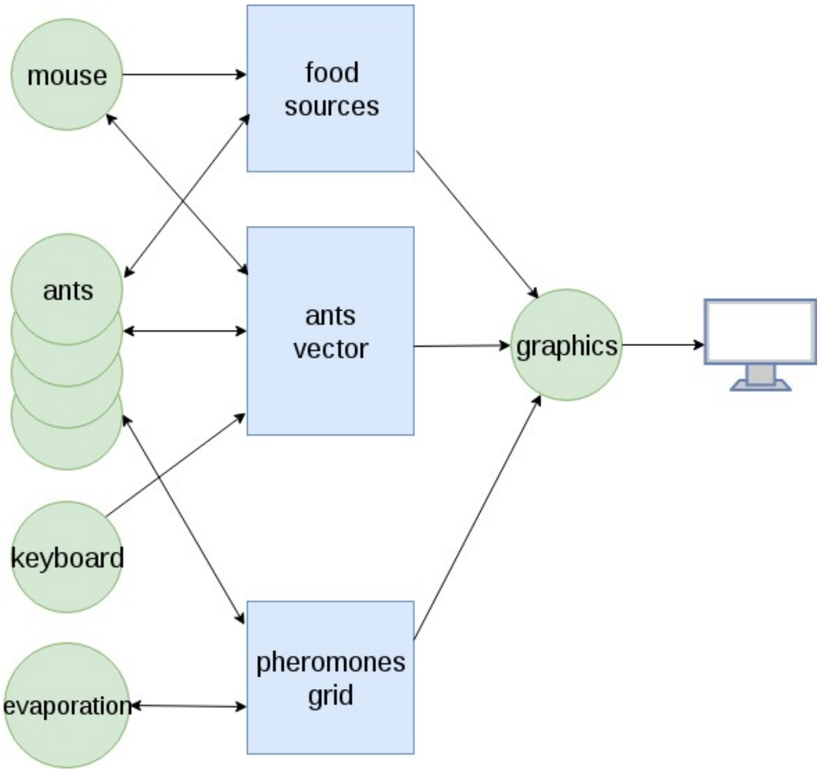

Figure 5: Tasks-resources diagram

5 Simulation parameters

The user can configure and play around a bunch of parameters that affect the simulation behaviour in different ways. The following tables are meant to describe the different parameters and where they are used within the code, along with the range of admitted values and suggested ones.

5.1 Ants

Parameter	Description	Range	Suggested
POP_SIZE_MAX	Population limit size	$\mathbb{N}$	50
STEP_LENGTH	Distance covered after each step	$\mathbb{N}$	5
EXPL_CONE	Max angle deviation after each step	$\mathbb{R}$	0.3
OLFACTION_RADIUS	Olfactory detection radius	$\mathbb{N}$	3
VISION_RADIUS	Visual detection radius	$\mathbb{N}$	30
MAX_AUDACITY	Chance to switch to exploration mode	$\mathbb{R}$	0.005
EXPL_DURATION	Exploration duration	$\mathbb{N}$	8
DEFAULT_POP	Initial population size	$\mathbb{N}$	10

Table 2: Ants parameters

5.2 Field

Parameter	Description	Range	Suggested
FIELD_WIDTH	Width of the field	$\mathbb{N}$	640
FIELD_HEIGHT	Height of the field	$\mathbb{N}$	480
MAX_FOOD_SRC	Max number of food sources	$\mathbb{N}$	30
FOOD_UNITS	Initial number of food units	$\mathbb{N}$	100
HOME_X	Anthill x coordinate	$\mathbb{N}$	60
HOME_Y	Anthill y coordinate	$\mathbb{N}$	60

Table 3: Field parameters

5.3 Pheromones

Parameter	Description	Range	Suggested
CELL_SIZE	Size of pheromone cell	$\mathbb{N}$	10
SMELL_UNIT	Pheromone intensity when initially dropped	$\mathbb{R}$	100.0
SMELL_TRESH	Minimum detectable smell value	$\mathbb{R}$	1.0
DROP_BACKOFF	Backoff time to drop twice in same cell	$\mathbb{N}$	5
DROP_FACTOR	Drop discount factor γ	[0,1]	0.975
EVAPOR_FACTOR	Evaporation factor α	[0,1]	0.98

Table 4: Pheromones parameters

6 Experiments

6.1 Drop discount factor

The drop discount factor γ can be tuned with the parameter DROP_FACTOR. Due to the presence of a minimum detectable pheromones quantity threshold, changing the parameter basically controls the length of the trail (the higher γ, the longer the trail). Low values make it difficult to establish stable paths between the anthill and food sources, while too high values make it hard to discard unnecessarily long paths, thus slowing down the convergence. Experiments showed that a value of 0.975 behaves quite well in our scenario.

6.2 Evaporation factor

The evaporation factor α is tied to the parameter EVAPOR_FACTOR and determines the pheromones decay rate. High values of the parameter keep the pheromones there for a longer period of time. Too high values make it difficult to get rid of suboptimal paths, whereas too low values result in no usable information at all.

References

[1] Wikipedia, **Stigmergy**, https://en.wikipedia.org/wiki/Stigmergy pages 2

[2] Wikipedia, **Ant colony optmization algorithm**, https://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms pages 2

[3] Wikipedia, **Random walk**, https://en.wikipedia.org/wiki/Random_walk pages 4

[4] Wikipedia, **Allegro library**, https://liballeg.org/ pages 6

[5] Wikipedia, **POSIX Threads**, https://en.wikipedia.org/wiki/POSIX_Threads pages 6